

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Запорізький національний технічний університет

Кафедра ПЗ

МЕТОДИЧНІ ВКАЗІВКИ

до виконання курсового проекту
“Програмна реалізація сервісних додатків AutoCAD
засобами AutoLISP”
з дисципліни

“Системи автоматизованого проектування”

для студентів освітнього ступеня бакалавр
за спеціальністю
122 “Комп’ютерні науки та інформаційні технології”
денної форми навчання

2016

Методичні вказівки до виконання курсового проекту “Програмна реалізація сервісних додатків AutoCAD засобами AutoLISP” з дисципліни “Системи автоматизованого проектування” для студентів освітнього рівня бакалавр за спеціальністю Комп’ютерні науки та інформаційні технології денної форми навчання / Укл.: А.В. Пархоменко, М.О. Андреев, М.В. Калініна. – Запоріжжя: ЗНТУ, 2016. – 75 с.

Укладачі: А.В. Пархоменко, к.т.н. доцент каф. ПЗ,
М.О. Андреев, асистент;
М.В. Калініна, асистент.

Рецензент: Г.М. Шило, к.т.н., доцент кафедри ІТЕЗ.

Відповідальний
за випуск: С.О. Субботін, зав. каф. ПЗ, д.т.н., професор.

Затверджено
на засіданні кафедри
програмних засобів
Протокол № 1
від “16” серпня 2016 р.

Рекомендовано до видання
НМК факультету
комп’ютерних наук і
технологій
Протокол № 1
від “22” серпня 2016 р.

ЗМІСТ

Вступ.....	5
1 Мета та задачі курсового проекту	7
2 Основні етапи курсового проекту та терміни їх виконання	8
2.1 Вхідні дані до проекту	8
2.2 Послідовність виконання курсового проекту	8
3 Вимоги до оформлення і обсягу курсового проекту	10
4 Вимоги до змісту курсового проекту	11
4.1 Структура пояснювальної записки.....	11
4.2 Вимоги до побудови основної частини курсового проекту.....	12
5 Теоретичні відомості	14
5.1 Загальні відомості про мову LISP	14
<i>5.1.1 Символи та списки</i>	<i>14</i>
<i>5.1.2 Поняття функції</i>	<i>14</i>
5.2 Використання AutoLISP в середовищі AutoCAD.....	15
<i>5.2.1 Інтерпретатор AutoLISP</i>	<i>15</i>
<i>5.2.2 Функції, визначені користувачем.....</i>	<i>17</i>
<i>5.2.3 Функція використання текстових файлів</i>	<i>18</i>
<i>5.2.4 Доступ з AutoLISP до команд AutoCAD</i>	<i>18</i>
<i>5.2.5 Доступ до примітивів AutoCAD.....</i>	<i>20</i>
5.3 Приклади розробки LISP-програм.....	20
5.4 Методика створення сервісних бібліотек в САПР AutoCAD	49
<i>5.4.1 Розробка LISP-програм та створення слайдів.....</i>	<i>50</i>

5.4.2 Створення файлу бібліотеки слайдів	52
5.4.3 Створення файлу меню сервісної бібліотеки AutoCAD	52
5.4.4 Підключення файлу меню до середовища AutoCAD ...	55
5.4.5 Налаштування інтерфейсу користувача AutoCAD 2011	57
5.4.6 Підключення сервісного додатку до AutoCAD 2011 ..	59
Рекомендована література	66
Додаток А. Завдання на курсовий проект	67
Додаток Б. Функції LISP	71
Додаток В. Функції AutoLISP	74

ВСТУП

Сучасний рівень програмних і технічних засобів електронної обчислювальної техніки дозволяє перейти від традиційних ручних методів конструювання до нових інформаційних технологій з використанням ЕОМ, створювати сучасні системи автоматизації проектування (CAD/CAM-системи), які відповідають вимогам міжнародних стандартів (ISO, STEP та ін.) як з точки зору якості виконання конструкторсько-технологічної документації так і дотримання певних вимог.

В діалозі з ЕОМ можливо створювати конструкторські документи (креслення, схеми) як з використанням графічних примітивів (типу точка, відрізок, окружність та ін.), так і фрагментів раніше створених конструктивних елементів (графічних зображень стандартних виробів, типових та уніфікованих конструкцій, їх частин). При цьому моделі вищевказаних фрагментів можуть бути параметрично заданими. За допомогою завдання значень параметрів конструктор може змінити їх розміри та геометричну форму, розташувати в необхідному місці креслення під заданим кутом, що забезпечує багатоваріантність графічних зображень і, відповідно, креслень і схем. При такому підході використання комп'ютерної графіки не усуває креслення як основу конструювання, а ЕОМ використовується як “електронний кульман”.

Інший, найсучасніший підхід до автоматизації конструкторської діяльності базується на використанні тривимірних геометричних моделей проєктованих виробів, що забезпечує перехід на більш високий, якісний рівень конструювання. Але, обмеженням у використанні систем високого рівня (інтегрованих CAD/CAM/CAE/PDM-систем), які підтримують сучасні інформаційні технології проєктування, є їх висока вартість. Тому, системи середнього та низького рівня (AutoCAD, КОМПАС та інші) є більш розповсюдженими на українських підприємствах.

Незважаючи на те, що AutoCAD – потужна система, яка підтримує об'єктно-орієнтовану технологію проєктування, вона не може забезпечити рішення усіх задач, які зустрічаються у практичній діяльності та задовольнити усі вимоги конкретної проєктованої галузі.

Для розширення функціональних можливостей AutoCAD та його адаптації для різноманітного застосування можливо використовувати програмування.

Автоматизація проектування в AutoCAD може здійснюватись різними засобами: від елементарних пакетних файлів (скриптів) до сучасних мов програмування (AutoLISP, Visual Basic, C++).

AutoLISP – мова програмування для AutoCAD, діалект мови Lisp, у який додані функції доступу до об'єктів та таблиць AutoCAD. Основою мови LISP є робота зі списками, які можуть мати довільну довжину та включати елементи різної природи (числа, текстові рядки та ін.).

Мова AutoLISP дозволяє розвивати функції AutoCAD стосовно конкретних задач і вимог. За допомогою цієї мови можливо виконувати: розробку параметрично-керованих моделей; розробку користувальницького інтерфейсу систем-надбудов; роботу з графічною базою; додавання нових команд; створення власного середовища проектування.

Таким чином, розробка спеціалізованих сервісних бібліотек, які містять зображення стандартних виробів, типових та уніфікованих деталей є актуальною задачею фахівця за спеціальністю 122 Комп'ютерні науки та інформаційні технології.

1 МЕТА ТА ЗАДАЧІ КУРСОВОГО ПРОЕКТУ

Курсовий проект з дисципліни "Системи автоматизації проектування" виконується студентами денної форми навчання спеціальності 122 Комп'ютерні науки та інформаційні технології в 6 (весняному) семестрі.

Метою курсового проекту є програмна реалізація сервісних бібліотек в середовищі AutoCAD.

Задачі курсового проекту:

- дослідити структурні та функціональні особливості САПР AutoCAD;
- вивчити засоби AutoLISP для розширення функціональних можливостей AutoCAD;
- вивчити методику створення сервісних додатків в середовищі AutoCAD;
- розробити структуру бібліотеки стандартних елементів;
- виконати програмну реалізацію сервісної бібліотеки в середовищі AutoCAD;
- розробити poradnik для користувача та для програміста.

2 ОСНОВНІ ЕТАПИ КУРСОВОГО ПРОЕКТУ ТА ТЕРМІНИ ЇХ ВИКОНАННЯ

2.1 Вхідні дані до проекту

Вхідною інформацією до виконання проекту є дані держстандартів на стандартні кріпильні деталі, які використовуються у машинобудуванні. Вхідні дані до проекту в залежності від варіанту містяться в додатку А.

2.2 Послідовність виконання курсового проекту

Основні етапи курсового проекту (табл. 2.1) мають бути виконані до кінця наступних тижнів семестру:

Таблиця 2.1 – Основні етапи курсового проекту

Етапи	Тижні
Аналіз вхідних даних	2
Вивчення особливостей мови AutoLISP	4
Опрацювання методики розробки LISP-програм	5
Програмна реалізація параметричної моделі заданого стандартного елемента	6
Вивчення методики створення сервісної бібліотеки	7
Розробка структурної схеми сервісної бібліотеки	8
Програмна реалізація сервісної бібліотеки в середовищі САПР AutoCAD	9
Оформлення пояснювальної записки до курсового проекту	10
Оформлення графічної частини	11
Попередня перевірка проекту керівником, рецензування	12
Захист курсового проекту	13

Контроль за виконанням курсового проекту здійснює керівник відповідно з трудомісткістю етапів проектування, графіком учбового процесу і графіком рубіжних контролів, що встановлює деканат. Для успішного виконання студентами курсового проекту керівник призначає консультації, відповідно з планами учбового навантаження. Захист курсових проектів здійснюється згідно з затвердженням

графіком. Порушення студентами етапності виконання курсового проекту та зриви графіку захистів недопустимі. За тиждень до призначеної дати захисту студент зобов'язаний пред'явити своєму керівнику пояснювальну записку та плакати для оцінки готовності і рецензування. Обов'язковою є перевірка на відповідність роботи вимогам СТП 15-96 “Пояснювальна записка до курсових та дипломних проектів. Вимоги та правила оформлення”.

Прийом курсового проекту здійснюється кафедральною комісією у складі не менше двох викладачів.

Захист передбачає доповідь з використанням мультимедійної презентації, під час якої студент повинен проявити глибокі теоретичні знання і практичні навички в постановці задачі та її програмній реалізації для досягнення мети проекту.

3 ВИМОГИ ДО ОФОРМЛЕННЯ І ОБСЯГУ КУРСОВОГО ПРОЕКТУ

Пояснювальна записка (ПЗ) до курсового проекту виконується згідно з вимогами СТП 15-96 “Пояснювальна записка до курсових та дипломних проектів. Вимоги та правила оформлення”. ПЗ повинна мати обсяг 40-50 сторінок.

Наявність графічної частині проекту передбачає оформлення слайдів презентації. Слайди повинні в достатньому обсязі висвітлювати зміст роботи: вхідні дані, ескіз розташування допоміжних точок, схема алгоритму методики розробки бібліотеки стандартних елементів, структурна та/або функціональна схема створеної бібліотеки, схема алгоритму функціонування lisp-програми, робочі вікна з результатами розробки сервісного додатку AutoCAD. Слайди містять інформацію, що обов'язково є в пояснювальній записці. Студент використовує слайди при захисті роботи.

4 ВИМОГИ ДО ЗМІСТУ КУРСОВОГО ПРОЕКТУ

4.1 Структура пояснювальної записки

ПЗ складається з таких обов'язкових частин:

- титульний аркуш;
- завдання на курсовий проект;
- реферат;
- зміст;
- вступ;
- основна частина;
- висновки;
- перелік посилань;
- додатки.

ЗАВДАННЯ НА КУРСОВИЙ ПРОЕКТ оформлюється на стандартному бланку.

РЕФЕРАТ повинен містити опис параметрів пояснювальної записки (кількість сторінок, рисунків, таблиць, додатків, використаних джерел); об'єкт розробки, мету та задачі проекту; результат розробки; новизну роботи; практичну цінність виконаних розробок; ключові слова (10-14 основних термінів роботи).

У **ВСТУПІ** має бути висвітлений сучасний стан задачі, що вирішується; визначена галузь проектування; наведена коротка інформація про систему автоматизованого проектування AutoCAD; визначена актуальність розробки сервісних спеціалізованих бібліотек в середовищі AutoCAD, описана мета курсового проекту.

У **ВИСНОВКАХ** мають бути наведені результати розробок, їх практичне значення та можливості подальшого використання.

ПЕРЕЛІК ПОСИЛАНЬ оформлюється згідно з вимогами СТП15-96. **Обов'язковими** є посилання на всі (у тому числі електронні) використані джерела у тексті пояснювальної записки.

У **ДОДАТКИ** необхідно винести програмну документацію: листи затвердження, специфікацію, опис програми, текст програми.

Програмний документ **ТЕХНІЧНЕ ЗАВДАННЯ** складається згідно з вимогами ГОСТ 19.201-78 «Технічне завдання. Вимоги до змісту та оформленню».

Програмний документ **СПЕЦИФІКАЦІЯ** складається згідно з вимогами ГОСТ 19.202-78 «Спецификация. Требования к содержанию и оформлению».

Програмний документ ОПИС ПРОГРАМИ складається згідно з вимогами ГОСТ 19.402-78 «Описание программы».

Програмний документ ТЕКСТ ПРОГРАМИ складається згідно з вимогами ГОСТ 19.401-78 «Текст программы. Требования к содержанию и оформлению».

Також в додатках наводяться слайди презентації.

4.2 Вимоги до побудови основної частини курсового проекту

Основна частина курсового проекту в загальному випадку має наступний вигляд.

1 Аналіз сучасного стану та перспектив розвитку CAD/CAM-систем.

1.1 Огляд сучасних САПР, що використовуються у машинобудівній галузі.

1.2 Структура, призначення та функціональні можливості САПР AutoCAD.

2 Розширення функціональних можливостей AutoCAD засобами AutoLISP.

2.1 Особливості мови LISP.

2.2 Мова AutoLISP як засіб адаптації AutoCAD до різноманітних застосувань.

3 Програмна реалізація спеціалізованих сервісних бібліотек в середовищі AutoCAD.

3.1 Аналіз вхідної інформації до проекту.

3.2 Схема алгоритму методики створення сервісної бібліотеки стандартних елементів.

3.3 Розробка структури сервісного додатку.

3.4 Опис особливостей програмної реалізації бібліотеки.

3.5 Порадник для користувача

3.6 Порадник для програміста.

Перший розділ повинен містити огляд сучасних систем автоматизації проектування, що використовуються у машинобудуванні. Шляхом аналізу літературних джерел, а також інформації з мережі Інтернет перелічити та дати загальну характеристику існуючих CAD/CAM-систем, висвітлити їх недоліки та переваги. Описати призначення, галузь застосування, характеристики, структуру, функціональні можливості актуальної версії САПР AutoCAD. В цьому розділі також необхідно обов'язково відобразити передумови та доцільність розробки сервісних спеціалізованих бібліотек в середовищі AutoCAD.

Другий розділ присвячений аналізу можливостей адаптації середовища AutoCAD для різноманітного застосування та урахування специфіки виробництва. Необхідно скласти класифікацію функцій мови LISP, навести основні відмінності та особливості мови AutoLISP. Висвітлити сутність технології багатоваріантного конструювання.

Третій розділ присвячується вирішенню практичних задач, пов'язаних з програмною реалізацією сервісної бібліотеки стандартних елементів. Перш за все, необхідно навести та проаналізувати вхідну інформацію до проекту: креслення заданого стандартного елемента (виконати в AutoCAD, проставити позначення розмірів) та параметричні дані (таблиці розмірів). Далі, навести схему алгоритму методики розробки сервісної бібліотеки. Схема повинна відображати всі етапи процесу і дозволяти отримати закінчене програмне рішення. Схема алгоритму оформлюється згідно з вимогами Держстандартів та ГОСТ 19.701-90 «Схемы алгоритмов, программ данных и систем». Обов'язковим є поблочний детальний опис схеми алгоритму.

Обов'язковою є наявність у ПЗ структурної схеми бібліотеки з детальним описом усіх її складових та посиланнями на тексти програм у додатку.

Далі, необхідно висвітлити особливості програмної реалізації сервісної бібліотеки: навести схему алгоритму функціонування програми, опис функцій AutoLisp та команд AutoCAD, що використовувались при створенні програми, ескіз розташування допоміжних точок, опис системних змінних. Обов'язковою є наявність коментарів у текстах програм.

Необхідно обґрунтувати вибір типу інтерфейсу програми (або класичний інтерфейс з командного рядка, або графічний за рахунок розробки додаткових компонентів для опису та забезпечення функціонування діалогових вікон). При розробці інтерфейсу програми забезпечити максимально можливу за умов вищезначеного вибору ефективну діалогову взаємодію з користувачем. У пояснювальній записці навести робочі вікна сервісного додатку та результати роботи програми.

Документ “Порадник для програміста” повинен містити перелік дій, які необхідно виконати програмісту для настройки, підключення та поповнення сервісної бібліотеки.

Документ “Порадник для користувача” повинен містити докладний, достатньо проілюстрований перелік дій конструктора для практичного застосування сервісної бібліотеки.

5 ТЕОРЕТИЧНІ ВІДОМОСТІ

5.1 Загальні відомості про мову LISP

5.1.1 Символи та списки

Для зображення об'єктів у мові LISP використовуються символи та побудовані з них символічні структури.

Символ – це ім'я, складене з літер, цифр та спеціальних знаків, що визначає предмет, об'єкт, річ, дію з реального світу. У LISP символи визначають числа, інші символи чи більш складні структури – програми (функції) та інші об'єкти мови.

У LISP використовуються три види констант: числові, рядкові та логічні. Вони не можуть зображати інші об'єкти LISP, крім самих себе чи свого власного значення.

Атоми – найпростіші об'єкти LISP, з яких будуються інші структури; вони представляють собою символи чи константи. Упорядкована послідовність, елементами якої є атоми чи списки (підсписки), називається *списком* (list). Списки завжди записуються у круглих дужках, елементи розділяються пропусками. Списки – це багаторівнева структура даних, у яких відкриті й закриті дужки знаходяться у суворій відповідності. Список, у якому немає жодного елемента, називається пустим списком та зображується як $()$ чи символом *NIL*. Пустий список – це не “нічого”. Він виконує ту саму роль, що і нуль у арифметиці. *NIL* може бути, наприклад, елементом інших списків. Атоми і списки – це основні типи даних мови LISP, вони називаються символічними виразами або S-виразами.

У LISP імена символів, змінних, списків, функцій та інших об'єктів не закріплені попередньо за якимись типами даних. Типи, взагалі, не зв'язані з іменами об'єктів даних, а супроводжують самі об'єкти. У цьому розумінні LISP є безтиповою мовою.

5.1.2 Поняття функції

У LISP як для виклику функції, так і для запису виразів прийнята однакова префіксна форма запису обчислень (Reverse Polish Notation – зворотна польська нотація), при якій ім'я функції записується спочатку, а далі записуються її аргументи, все всередині

дужок. При обчисленні значення функцій не треба здійснювати складний синтаксичний аналіз виразів, оскільки вже по першому символу поточного виразу система розпізнає, з якою структурою має справу та як її інтерпретувати або як із нею поводитися.

Програми будуються із логічно розділених визначень функцій. Визначення складаються із керуючих структур, що організовують обчислення, та з вкладених, що часто викликають самих себе (рекурсивних), викликів функцій. Основними засобами функціонального програмування є композиція та рекурсія.

LISP працює у режимі інтерпретації. Програми немає необхідності транслювати та можливе їх виправлення в процесі виконання.

У додатку Б наведені функції LISP, які рекомендовано використовувати при розробці сервісного додатку.

5.2 Використання AutoLISP в середовищі AutoCAD

5.2.1 Інтерпретатор AutoLISP

У склад AutoCAD входить інтерпретатор AutoLISP, який починає працювати, якщо у командному рядку у відповідь на стандартний запит **Команда: (Command:)** введено відкриваючу дужку. У цю мить AutoCAD настраюється на прийом та обчислення виразу AutoLISP, який повинен мати форму списку, тобто починатися відкритою дужкою та закінчуватися закритою дужкою, а елементи списку повинні відділятися один від одного пропусками.

(<функція [<arg1> [<arg2> ... [<argN>] ...])

Наприклад, для одержання суми трьох чисел ви можете ввести рядок як показано на рис. 5.1:

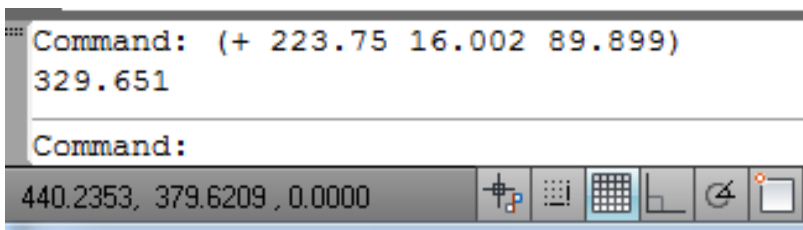


Рисунок 5.1 – Приклад роботи інтерпретатора Lisp

Як тільки завершено введення виразу натисненням на клавішу **<Enter>**, система AutoCAD обчислює його та видає результат.

Кількість аргументів залежить від синтаксису функції. Існують функції, які викликаються без аргументів.

З точки зору AutoLISP все, що розташоване у круглих дужках, є списком, який треба обробити як обчислюваний вираз.

Основне правило AutoLISP – баланс дужок, тобто скільки дужок відкрито, стільки повинно бути й закрито. У інтегрованому середовищі розробки Visual LISP передбачені механізми перевірки дужок у файлах з текстами програм.

У якості будь-якого аргументу функції може бути введений обчислюваний вираз.

Інтерпретатор AutoLISP обчислює значення кожного введеного виразу та повертає його у середовище AutoCAD. Повернене значення, у свою чергу, або може бути передане для наступних обчислень у вираз більш високого рівня, або може бути збережене у змінній за допомогою функції групового присвоювання **SETQ**, яка має наступний синтаксис:

(SETQ <змінна1> <вираз1> [... [<зміннаN> <виразN>]])

Наприклад:

(SETQ A (+ 223.75 16.002 89.899))

У подальшому використати значення змінної **A** можливо за допомогою префіксу **“!”** (знак оклику). Якщо у командному рядку системи AutoCAD у відповідь на стандартну підказку **Команда:** (**Command:**) ввести **!A**, то система видасть поточне значення змінної (рис. 5.2)

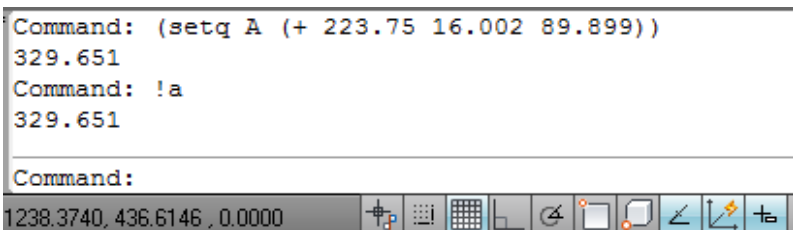


Рисунок 5.2 – Приклад звертання до змінної

У текст програми корисно вставляти коментарі. Коментарем вважається частина рядка, починаючи з символу **“;”** (крапка з комою) і

закінчуючи останнім символом рядка. Якщо треба вставити багаторядковий коментар, починаючи з середини рядка, тоді його треба розмістити між символами “;|” та “|;”.

5.2.2 Функції, визначені користувачем

LISP-програми можуть бути використані для автоматизації будь-якої зі звичайних задач, які повторюються, або для розширення функціональних можливостей AutoCAD. Наприклад, якщо приходится багаторазово виконувати однакову послідовність команд, може виявитися більш ефективним витратити деякий час на написання функцій AutoLISP, які автоматизують виконання задачі, або створюють новий метод її вирішення.

Також відомо, що розробка конструкцій і креслень здебільшого полягає у їх формуванні з однотипових, уніфікованих чи стандартних елементів. Тому, доцільно для цих елементів розробляти функції чи команди, які інтегруються до AutoCAD. Звертання до таких команд можливо організовувати, використовуючи спадаючі та графічні меню. Для цього необхідно створити власний інтерфейс, що забезпечує роботу з комп'ютером у звичних для конструктора термінах, визначеннях і зображеннях.

Функція – це команда чи ряд команд, які AutoCAD виконує у визначеному порядку. Кожна функція повинна мати ім'я, щоб інші функції мали можливість викликати її, вказуючи це ім'я.

AutoLISP притаманна природна розширюваність за рахунок визначення нових функцій. Це дає користувачу можливість визначити власні функції за допомогою функції **DEFUN** (define function):

```
(DEFUN <ім'я_функції>
  (|<формальні_аргументи>|
  |<локальні_змінні>|)
  <тіло_функції>
```

),

де <ім'я_функції> – будь-який символ;

<формальні_аргументи> – послідовність символів розділених пропусками. Це аргументи, які не втрачають свої значення після закінчення роботи функції;

<локальні_змінні> – символи, що використовуються для позначення інших об'єктів AutoLISP тільки всередині визначення функції. Вони активні тільки при роботі конкретної функції.

<тіло_функції> – один/декілька S-виразів, тобто <S-вираз>. Функція повертає значення останнього обчисленого S-виразу.

5.2.3 Функція використання текстових файлів

Клавіатурне введення виразів AutoLISP використовується рідко. Зазвичай вирази AutoLISP записуються у текстовий файл з розширенням **.lsp*, який потім завантажується у AutoCAD за допомогою функції **LOAD**:

(LOAD <ім'я_файла> [<код_помилки>])

Наприклад, щоб завантажити файл з ім'ям *myfile.lsp*, треба задати:

(LOAD "myfile")

Інтерфейс AutoCAD надає можливість завантажити файли з визначеннями функцій, використовуючи діалогові вікна. Для цього треба зробити наступне (для AutoCAD 2011):

а) виберіть з меню **Tools** (Інструменти) в AutoCAD Classic (або **Manage** в всіх середовищах AutoCAD2011, наприклад 2D Drafting & Annotation) пункт **Load Application...** (Завантаження додатку...), щоб відкрити діалогове вікно **Load\Unload Applications** (Завантаження/Вивантаження додатків);

б) виберіть файл LISP-програми, який треба завантажити;

в) натисніть кнопку **Load** в діалоговому вікні **Load\Unload Applications**

Для виклику цього меню також можна ввести команду **APpload** з командного рядку.

Зазвичай файл, що завантажується, містить визначення власних функцій, описаних з використанням **DEFUN**, хоча це може бути й звичайна послідовність дій, описаних на мові AutoLISP.

Якщо завантаження успішно завершено, **LOAD** повертає ім'я останньої функції, визначеної у файлі, якщо ні – повертає ім'я файлу.

5.2.4 Доступ з AutoLISP до команд AutoCAD

Для перетворення пакетного файлу команд AutoCAD в програму на мові LISP всі команди переводять в вирази на цій мові, наприклад:

Скрипт	CIRCLE 210,180 D 45
Вираз на Lisp	(COMMAND "CIRCLE" ' (210 180) "D" 45 "")

Функція **COMMAND** забезпечує доступ до команд AutoCAD.
(**COMMAND [аргументи]...**)

Аргументи – це команди AutoCAD, їх ключі, координати точок, числові та текстові параметри. Кожний аргумент обчислюється та посилається у редактор AutoCAD як відповідь на конкретний запит. Команди та ключі наводяться як рядкові константи (наприклад, **"CIRCLE"**, **"TTR"**).

Увага! Щоб уникнути помилок у програмах, для запису рядкових констант необхідно використовувати звичайні лапки (не парні!). Це має бути символ " (з кодом 34 або 22h).

Точки можливо задавати і як рядкові константи (наприклад, "10,10"), і як списки з двох-трьох дійсних чисел (наприклад, (LIST 10.0 (+ 10.0 10.0))).

Приклад:

```
(SETQ PT1 (LIST 10.0 10.0))
(SETQ PT2 (LIST 20.0 20.0))
(COMMAND "LINE" PT1 PT2 "")
```

Порожній рядок (") дорівнює натисненню клавіш <Space> або <Enter> на клавіатурі.

Якщо ви не впевнені, чи ваша програма буде працювати з російською версією AutoCAD, то слід використовувати англійські імена команд з підкресленням на початку (наприклад, LINE). З такими іменами команди виконуються у будь-якій версії системи.

Зчитування та завдання системних змінних AutoCAD в AutoLISP здійснюють функції **GETVAR**, **SETVAR**, **GETENV**. Пошук файлу з заданим ім'ям здійснюють функції **FINDFILE**, **GETFILED**. Застосовуються також функції геометричних обчислень: **ANGLE**, **DISTANCE**, **INTERS**, **POLAR**, **OSNAP** (див. додаток В).

Теоретично можливо обмежитися лише простими типами даних LISP: атомарними об'єктами (числами, символами, логічними значеннями), списками та засобами їх внутрішнього представлення. Але це має такі ж недоліки, як і будь-яке програмування з використанням тільки базових функцій. Тому, для введення даних різних типів використовуються наступні функції: **GETANGLE**, **GETORIENT**, **GETPOINT**, **GETCORNER**, **GETDIST**, **GETINT**, **GETKEYWORD**, **GETREAL**, **GETSTRING** (див. додаток В).

5.2.5 Доступ до примітивів AutoCAD

Інструментом доступу до примітивів AutoCAD є два спеціальних типи даних – ім'я примітива та набір. Вони керуються спеціальними функціями. Ім'я примітива – це покажчик в файлі, що створений редактором AutoCAD. Він дозволяє знайти запис бази даних AutoCAD, в якому міститься інформація про примітив. Набір – це об'єднання декількох імен примітивів. Імена примітивів та набори діють лише під час поточного сеансу роботи редактора AutoCAD. Використовуються функції імені примітива, маніпуляції набором примітивів, оперування даними примітивів креслення (див. додаток В).

5.3 Приклади розробки LISP-програм

Приклад 1. Розглянемо приклад програми, яка дозволяє накреслити елементарну деталь (рис. 5.3). Шар, у якому буде виконуватись креслення, створюється програмно, за допомогою команди **LAYER** та її ключа **MAKE** (створити новий шар та зробити його поточним).

Для запуску програми у командному рядку AutoCAD необхідно звернутися до функції **SAM**:

```
(SAM X Y H W),
```

де X, Y – координати базової точки;

H – висота фігури;

W – довжина фігури.

Наприклад, звертання до функції **SAM** з різними параметрами дає різні за розмірами зображення на рис. 5.3:

```
(SAM 0 5 15 20)
```

```
(SAM 28 5 4 4)
```

```
(SAM 40 5 10 17)
```

Текст програми:

```
; очищення поля креслення формату A4
(COMMAND "ERASE" "C" "0,0" "210,297" "")
(DEFUN SAM (X Y H W)
; визначення локальних змінних
(SETQ W1 (/ W 4.0))
```

```

(SETQ W2 (- W W1))
(SETQ W3 (/ W 2.0))
; збільшення області майбутнього креслення
; щоб усунути хибне спрацьовування прив'язок
(COMMAND "ZOOM" "W"
  (LIST (- X 5) (- Y 5))
  (LIST (+ X W 5) (+ Y H 5))
)
; створення поточного шару OSN
(COMMAND "LAYER" "MAKE" "OSN" "")
(COMMAND "PLINE"
  (LIST X Y)
  (LIST X (+ Y H))
  (LIST (+ X W1) (+ Y H))
  ; режим дуги (за центром та кінцевою точкою)
  "A" "CE"
  (LIST (+ X W3) (+ Y H))
  (LIST (+ X W2) (+ Y H))
  "L"
  (LIST (+ X W) (+ Y H))
  (LIST (+ X W) Y)
  "C"
)
; розміщення на екрані всього зображення
(COMMAND "_ZOOM" "_E")
)

```

Результати роботи програми наведені на рис. 5.3.

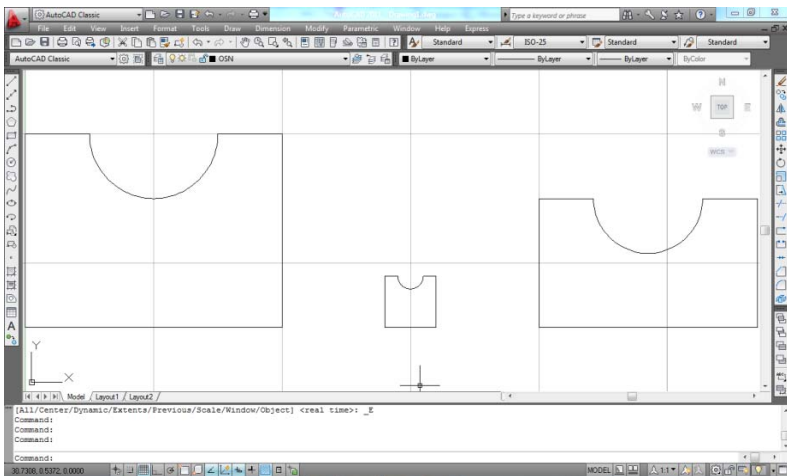


Рисунок 5.3 – Результати роботи програми для прикладу 1

Функція LISP – **LIST** дозволяє формувати координати точки, тобто це список, у якому перше число – це координата точки по X, друге – по Y, третє – по Z.

У програмі були використані наступні команди AutoCAD:

PLINE – креслення полілінії (у програмі ця команда використовує режим дуг та відтинків);

LAYER – використовується для операцій з шарами (в даному випадку – створення поточного шару OSN).

Треба звернути увагу на операцію ділення – якщо обидва операнди явно задані як цілі числа, то результат також буде цілим. Щоб уникнути помилок креслення рекомендується константи задавати явно дійсного типу (з точкою та нулем).

Приклад 2. Розглянемо більш складну програму побудови креслення у двовимірному просторі середовища AutoCAD (рис.5.4).

Для запуску програми у командному рядку AutoCAD необхідно звернутися до функції VAR5:

```
(VAR5 S H N X Y) ,
```

де S – довжина виступів;

H – висота виступів та відстань між ними;

N – кількість виступів;

X, Y – розміщення першого виступу на кресленні.

Функцію можливо викликати з різними параметрами, що дає різні за розміром зображення на рис. 5.4. Наприклад:

```
(VAR5 10 10 4 70 170)
(VAR5 30 10 6 140 170)
```

Текст програми:

```
; очищення поля креслення формату A4
(COMMAND "ERASE" "C" "0,0" "210,297" "")
; створення поточного шару OSN
(COMMAND "LAYER" "MAKE" "OSN" "")
; прив'язка тільки за кінцевою точкою
(SETVAR "OSMODE" 1)
(DEFUN VAR5 (S H N X Y)
  (SETQ X1 (- X 20) ; x центру кола
```

```

        Y1 (- Y 130) ; у центру кола
        I 1)          ; лічильник кількості виступів
; збільшення області майбутнього креслення
; щоб усунути хибне спрацьовування прив'язок
(COMMAND "ZOOM" "W"
  (LIST (+ X 5 S) (+ Y 5))
  (LIST (+ X 20 15 5) (- Y 130 15 5)))
; початок полілінії у нижній точці останнього виступу
(COMMAND "PLINE"
  (LIST X (- Y (* N H 2.0)))
  (LIST X Y1)          ; початок дуги
  "Arc"                ; режим дуг
  (LIST X1 (- Y 110)) ; кінець дуги
  "Line"               ; режим відтинків
  (LIST X1 Y)          ; лівий верхній кут
  (LIST X Y)          ; початок 1-го виступу
)
; цикл креслення виступів
(WHILE (<= I N)
; обчислення зміщення виступу за віссю Y
  (SETQ DY (* -2.0 H (1- I)))
; креслення виступу
  (COMMAND "LINE"
    (LIST (+ X S) (+ Y DY))
    (LIST (+ X S) (+ Y (- DY H)))
    (LIST X (+ Y (- DY H)))
    (LIST X (+ Y (- DY (* 2.0 H))))
  )
; збільшення лічильника виступів
  (SETQ I (1+ I))
)
; завершення виконання команди LINE
(COMMAND "")
; креслення кола
(COMMAND "CIRCLE" (LIST X1 Y1) 15.0)
)

```

У програмі були використані наступні команди AutoCAD:

PLINE – використовується для формування двовимірної полілінії (у режимах дуг та відтинків);

LINE – використовується для формування ламаної лінії;

CIRCLE – креслення кола за координатами центру та радіусом.

Також було модифіковано значення системної змінної **OSMODE**, яка містить бітові поля з режимами прив'язок. Щоб ввімкнути певні прив'язки, необхідно присвоїти цій змінній суму значень, що відповідають необхідним прив'язкам (див. табл. 5.1).

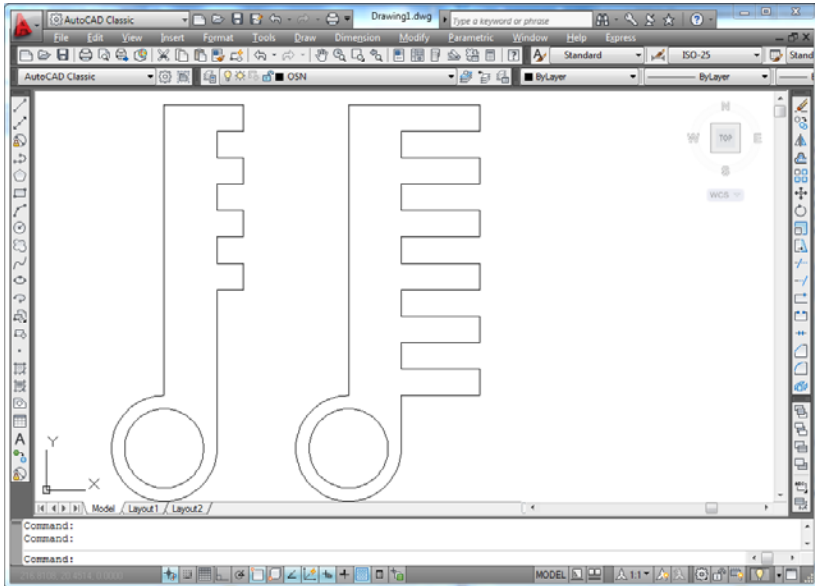


Рисунок 5.4 – Результати роботи програми для прикладу 2

Таблиця 5.1 – Режими об'єктних прив'язок

Код	Назва	Значення
0	NONE	вимкнути прив'язки
1	ENDpoint	за кінцевою точкою
2	MIDpoint	за точкою середини
4	CENter	за центром
8	NODe	за точкою (POINT)
16	QUAdrant	за квадрантними точками кіл, дуг, еліпсів
32	INTersection	за перетином
64	INSertion	за точкою вставки блока/тексту
128	PERpendicular	перпендикулярно
256	TANgent	за дотичною
512	NEArest	до найближчої точки
1024	Clears all object snaps	вимкнути всі прив'язки

Продовження таблиці 5.1

2048	APParent Intersection	за уявним перетином
4096	EXTension	за тимчасовим продовженням дуги або лінії
8192	PARallel	паралельно

Другий, більш наочний метод ввімкнення прив'язок полягає у виклику команди **OSNAP** з переліком перших трьох букв імен потрібних прив'язок, розділених комами:

(COMMAND "OSNAP" "END,INT")

Третій метод використання прив'язок полягає у глобальному відключенні прив'язок, та запису імені потрібної прив'язки безпосередньо перед точкою. Наприклад:

(COMMAND "LINE"

"END" PP1

"INT" (LIST X Y))

Слід зауважити, що на функціонування механізму прив'язок значний вплив має системна змінна **APERTURE** (за замовчанням дорівнює 10) – це радіус спрацьовування прив'язки у пікселях. Тому, якщо типові екранні розміри креслення близькі до значення цієї змінної, та якщо ввімкнені прив'язки, існує велика імовірність хибного спрацьовування. Для її усунення рекомендується збільшити область, яку займатиме майбутнє креслення, на весь екран (команда **ZOOM** з ключем **Window**). У насичених мілкими примітивами областях креслення доцільно застосувати їх додаткове збільшення.

Також подібний вплив має ввімкнення режиму прив'язки за сіткою (**SNAP**). Тому, при програмному виконанні креслення рекомендується її вимкнути.

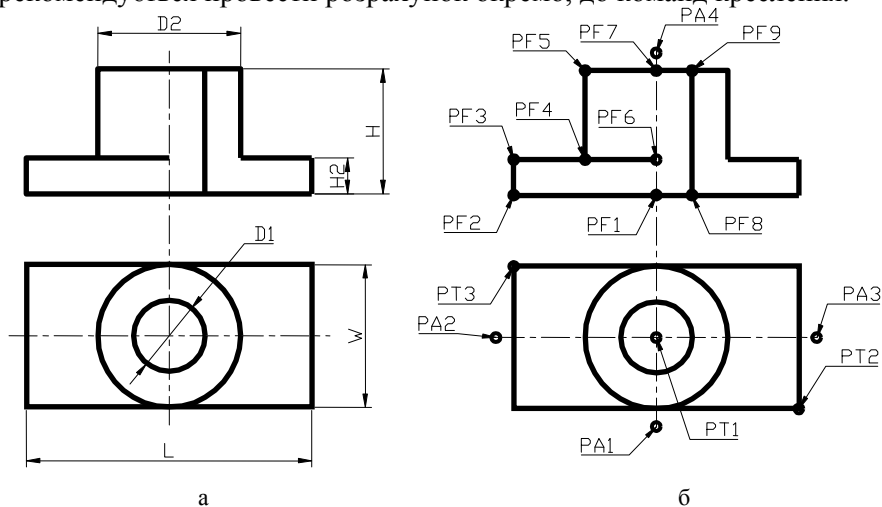
Приклад 3. Наведемо приклад побудови основи комплексної деталі у двовимірному вигляді, яка часто зустрічається у практиці моделювання деталей у машинобудівній галузі (рис. 5.5a).

Для більшої наочності подальших розрахунків потрібно проаналізувати цей ескіз, розбити його на мінімальну кількість примітивів та операцій, виявити потрібну множину прив'язок, накреслити ескіз розташування допоміжних точок, за якими буде виконане креслення (рис. 5.5б) та обрати базову точку для видів (точка PT1).

Зважаючи на вид креслення, потрібно забезпечити програмне створення двох шарів – для основних товстих ліній та для осевих тонких. Для цього використовуються наступні ключі команди **LAYER**:

- **Make** – створити новий шар та зробити його поточним;
- **Set** – встановити вказаний шар поточним;
- **New** – створити новий шар зі вказаним іменем;
- **Color** – задати колір примітивів вказаних шарів;
- **Ltype** – задати тип лінії примітивів вказаним шаром;
- **LWeight** – задати товщину ліній вказаних шарів.

Через достатньо складне креслення (велику кількість його параметрів) програма міститиме громіздкі розрахунки значень змінних, тому для забезпечення читабельності програми рекомендується провести розрахунок окремо, до команд креслення.



а – ескіз видів; б – ескіз розташування допоміжних точок

Рисунок 5.5 – Комплексна деталь

Функція користувача буде мати наступний формат виклику (параметри відповідно до позначень на рис. 5.5a):

(DETAIL3 X Y L W D1 H H2)

де X, Y – координати базової точки;

L – довжина основи деталі;

W – ширина основи деталі;

D1 – внутрішній діаметр циліндру;

H – висота деталі;

H2 – висота основи деталі.

Наприклад, звернення до функції **DETAIL3** з різними параметрами, дає результати роботи, наведені на рис. 5.6:

(DETAIL3 100 100 160 80 40 70 20)

(DETAIL3 360 100 320 60 40 50 10)

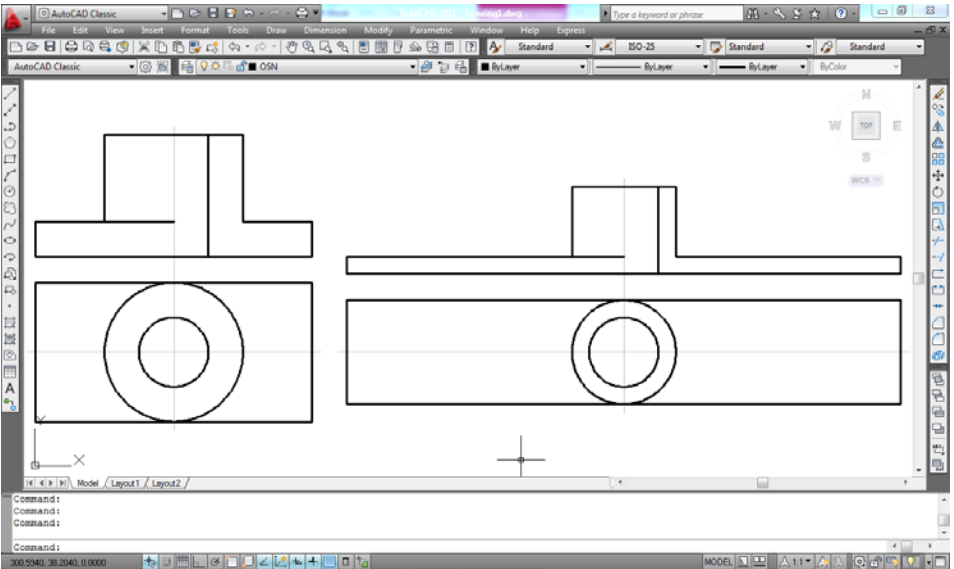


Рисунок 5.6 – Результати роботи програми для прикладу 3.

Текст програми:

```
; очищення поля креслення
(COMMAND "ERASE" "ALL" "")
; ввімкнення режиму відображення товщини ліній
(SETVAR "LWDISPLAY" 1)
; створення шарів OSI та OSN
(COMMAND "LAYER" "NEW" "OSI"
          "COLOR" "9" "OSI"
          "LTYPE" "CENTER" "OSI"
          "LWEIGHT" "0.15" "OSI" "")
(COMMAND "LAYER" "NEW" "OSN"
          "COLOR" "7" "OSN")
```

```

        "LTYPE" "CONTINUOUS" "OSN"
        "LWEIGHT" "0.30" "OSN" "")
; вимкнення об'єктних прив'язок
(COMMAND "OSNAP" "OFF")
; вимкнення прив'язки за сіткою
(COMMAND "SNAP" "OFF")
(DEFUN DETAIL3 (X Y L W D1 H H2 /
                W2 L2 R2 R1 DV AE
                PA1 PA2 PA3 PA4 PT1 PT2 PT3
                PF1 PF2 PF3 PF4 PF5 PF6 PF7 PF8)
; визначення локальних змінних
(SETQ W2 (/ W 2.0)
      L2 (/ L 2.0)
      R1 (/ D1 2.0)
      DV 15 ; вертикальна відстань між видами
      AE 5   ; виступ осьової лінії за основну
; розрахунок допоміжних точок
      PA1 (LIST X (- Y W2 AE))
      PA2 (LIST (- X L2 AE) Y)
      PA3 (LIST (+ X L2 AE) Y)
      PA4 (LIST X (+ Y W2 DV H AE))
      PT1 (LIST X Y)
      PT2 (LIST (+ X L2) (- Y W2))
      PT3 (LIST (- X L2) (+ Y W2))
      PF1 (LIST X (+ Y W2 DV))
      PF2 (LIST (- X L2) (+ Y W2 DV))
      PF3 (LIST (- X L2) (+ Y W2 DV H2))
      PF4 (LIST (- X W2) (+ Y W2 DV H2))
      PF5 (LIST (- X W2) (+ Y W2 DV H))
      PF6 (LIST X (+ Y W2 DV H2))
      PF7 (LIST X (+ Y W2 DV H))
      PF8 (LIST (+ X R1) (+ Y W2 DV))
      PF9 (LIST (+ X R1) (+ Y W2 DV H))
; збільшення області майбутнього креслення
; щоб усунути хибне спрацьовування прив'язок
(COMMAND "ZOOM" "W"
        (LIST (- X L2 AE AE) (- Y W2 AE AE))
        (LIST (+ X L2 AE AE) (+ Y W2 DV H AE AE)))
; перехід до шару осьових ліній
(COMMAND "LAYER" "SET" "OSI" "")
(COMMAND "LINE" PA1 PA4 "")
(COMMAND "LINE" PA2 PA3 "")
; перехід до шару основних ліній
(COMMAND "LAYER" "SET" "OSN" "")
; креслення виду зверху
(COMMAND "RECTANG" PT2 PT3)
(COMMAND "CIRCLE" "INT" PT1 R1)

```

```
(COMMAND "CIRCLE" "INT" PT1 W2)
; креслення виду спереду
(COMMAND "PLINE" PF1 PF2 PF3 PF4 PF5 PF7 "")
(COMMAND "MIRROR"
  "INT" PF2 ""
  "INT" PF1 "INT" PF7
  "N")
(COMMAND "LINE" "END" PF4 PF6 "")
(COMMAND "LINE" "NEA" PF8 "NEA" PF9 "")
; збільшення існуючого креслення на весь екран
(COMMAND "ZOOM" "Extents")
)
```

У програмі були використані специфічні функції AutoLISP:

(SETVAR varname value) – встановлення значення системної змінної AutoCAD з іменем *varname*;

У програмі були використані наступні команди AutoCAD:

LAYER – створення та настройка шарів (ім'я, колір, тип та товщина лінії);

OSNAP – керування об'єктними прив'язками;

SNAP – керування прив'язкою за сіткою;

ZOOM – масштабування зображення креслення (збільшення рамкою – ключ **Window**; вписати все креслення на весь екран – ключ **Extents**);

LINE – креслення ламаної лінії;

RECTANG – креслення прямокутника за двома точками (діагональ);

CIRCLE – креслення кола за центром та радіусом;

PLINE – креслення полілінії;

MIRROR – операція дзеркального відображення вказаного примітиву симетрично вісі, визначеної двома точками.

Було змінено значення системної змінної **LWDISPLAY**, яка перемикає режим відображення товщини ліній (0 – відображати, 1 – не відображати).

Також були використані об'єктні прив'язки за перетином **"INT"**, за кінцевою точкою **"END"**, за найближчою точкою **"NEA"**.

Приклад 4. Розглянемо програму, яка на основі прикладу 3 була ускладнена функціональними елементами – фаскою та штриховкою (рис. 5.7а).

Для більшої наочності подальших розрахунків потрібно проаналізувати цей ескіз, розбити його на мінімальну кількість

примітивів та операцій, виявити потрібну множину прив'язок, накреслити ескіз розташування допоміжних точок, за якими буде виконане креслення (рис. 5.76) та обрати базову точку для видів (точка PT1).

Зважаючи на вид креслення, потрібно забезпечити програмне створення трьох шарів – для основних товстих ліній, для осьових тонких та для штриховки. Для цього використовуються наступні ключі команди **LAYER**:

- **Make** – створити новий шар та зробити його поточним;
- **Set** – встановити вказаний шар поточним;
- **New** – створити новий шар зі вказаним іменем;
- **Color** – задати колір примітивів вказаних шарів;
- **Ltype** – задати тип лінії примітивів вказаним шарам;
- **LWeight** – задати товщину ліній вказаних шарів.

Через достатньо складне креслення (велику кількість його параметрів) програма міститиме громіздкі розрахунки значень змінних, тому для забезпечення читабельності програми рекомендується провести розрахунок окремо, до команд креслення.

Також необхідно реалізувати перевірку взаємозалежних вхідних параметрів функції **DETAIL4**. Повинні виконуватись наступні умови:

$$\left\{ \begin{array}{l} WF < W/2 \\ D < W \\ WS < D \\ H2 < H \\ DS < H \\ H2 + DS < H \end{array} \right.$$

Функція користувача буде мати наступний формат виклику (параметри відповідно до позначень на рис. 5.76):

(DETAIL4 X Y W D WF H H2 DS WS)

де X, Y – координати базової точки;

W – довжина сторони квадратної основи деталі;

D – зовнішній діаметр циліндру;

WF – ширина фаски на ребрах основи;

H – висота деталі;

H2 – висота основи деталі;

DS – глибина шліца;

WS – ширина шліца та внутрішній діаметр циліндру.

Наприклад, звернення до функції **DETAIL4** з різними параметрами, дає результати роботи, наведені на рис. 5.8:

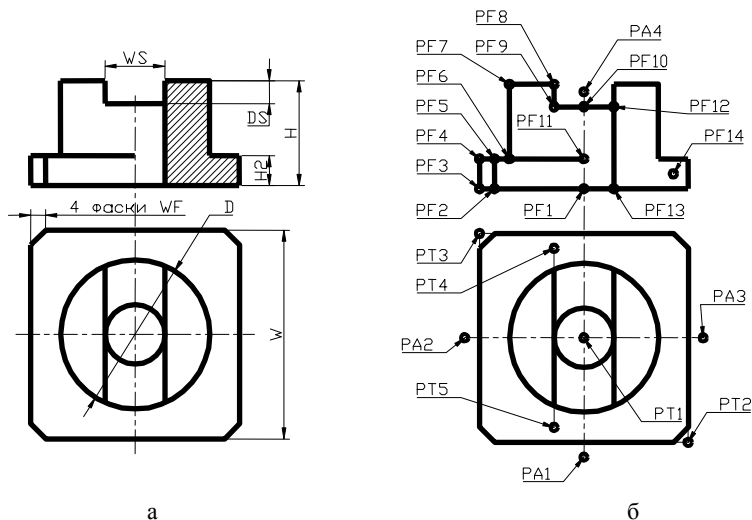
(DETAIL4 100 100 140 100 10 70 20 15 40)

(DETAIL4 275 100 100 50 10 100 20 30 30)

У програмі були використані наступні нові команди AutoCAD (порівняно з прикладом 3):

RECTANG – креслення прямокутника з фасками на кутах (ключ **Chamfer**) за двома точками (діагональ). Зверніть увагу, що вказані ненульові значення розмірів фасок – це глобальні зміни, які впливатимуть на креслення наступних прямокутників;

BHATCH – штриховка замкненої області з вказаною внутрішньою точкою за вказаним шаблоном (**ANSI31** – коса штриховка під кутом 45 градусів);



а – ескіз видів; б – ескіз розташування допоміжних точок

Рисунок 5.7 – Ускладнена комплексна деталь

У програмі були використані специфічні функції AutoLISP:

(PROGN [expr]...) – операторні дужки (виконання послідовності операцій **expr** у тому місці програми, де очікується одиночний вираз);

(ALERT string) – функція виводить діалогове вікно зі вказаним текстом **string** (повідомленням про помилку);

(EXIT) – використовується для примусового переривання виконання поточної функції та повернення до командного рядку;

(ENTLAST) – функція повертає внутрішній ідентифікатор останнього накресленого або модифікованого примітиву (використана для спрощення вибору примітиву в операції **TRIM**);

В залежності від типових розмірів креслення довжина штрихів штрих-пунктирних ліній може бути занадто велика/мала. Тому в даному прикладі було модифіковано значення системної змінної **LTSCALE** для зміни глобального масштабу штрихів ліній.

Також були використані об'єктні прив'язки за перетином **"INT"**, за кінцевою точкою **"END"**, за найближчою точкою **"NEA"**, за квадрантною точкою кола **"QUA"** (для уникнення конфлікту при виборі великого кола між цим колом та вертикальною віссю).

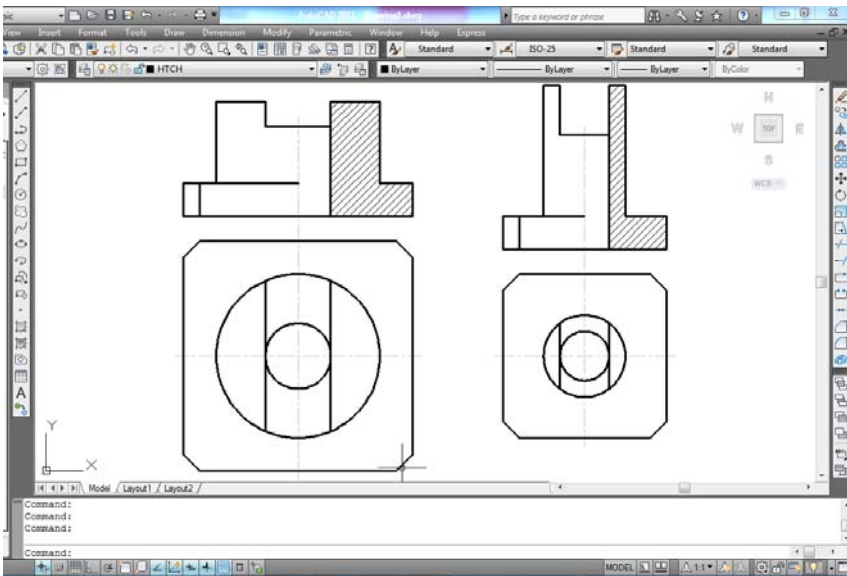


Рисунок 5.8 – Результати роботи програми для прикладу 4.

Текст програми:

```
; очищення поля креслення
(COMMAND "ERASE" "ALL" "")
; ввімкнення режиму відображення товщини ліній
```

```

(SETVAR "LWDISPLAY" 1)
; збільшення глобального масштабу штрихів ліній
(SETVAR "LTSCALE" 8)
; створення шарів OSI, OSN та HTCH
(COMMAND "LAYER" "NEW" "OSI"
      "COLOR" "9" "OSI"
      "LTYPE" "CENTER" "OSI"
      "LWEIGHT" "0.15" "OSI" "")
(COMMAND "LAYER" "NEW" "OSN"
      "COLOR" "7" "OSN"
      "LTYPE" "CONTINUOUS" "OSN"
      "LWEIGHT" "0.30" "OSN" "")
(COMMAND "LAYER" "NEW" "HTCH"
      "COLOR" "7" "HTCH"
      "LTYPE" "CONTINUOUS" "HTCH"
      "LWEIGHT" "0.15" "HTCH" "")
; вимкнення об'єктних прив'язок
(COMMAND "OSNAP" "OFF")
; вимкнення прив'язки за сіткою
(COMMAND "SNAP" "OFF")
(DEFUN DETAIL4
  (X Y W D WF H H2 DS WS
   / W2 R DV AE S1 WS2 RID SL
   PA1 PA2 PA3 PA4 PT1 PT2 PT3 PT4 PT5
   PF1 PF2 PF3 PF4 PF5 PF6 PF7
   PF8 PF9 PF10 PF11 PF12 PF13 PF14)
  ; визначення локальних змінних
  (SETQ W2 (/ W 2.0)
        R (/ D 2.0)
        DV 15 ; вертикальна відстань між видами
        AE 5 ; виступ осьової лінії за основну
        S1 (/ (- W D) 4.0)
        WS2 (/ WS 2.0)
        ; прапор правильності переданих даних побудови
        RID (AND (< WF W2) (< D W) (< WS D) (< H2 H)
                  (< DS H) (< (+ H2 DS) H)))
  ; перевірка залежних змінних
  (IF (EQUAL RID NIL)
    (PROGN
      ; вивід повідомлення про помилку
      (ALERT "Wrong input data!")
      (EXIT) ; вихід з функції
    )
  )
  ; розрахунок допоміжних точок
  (SETQ PA1 (LIST X (- Y W2 AE))
        PA2 (LIST (- X W2 AE) Y)

```

```

PA3 (LIST (+ X W2 AE) Y)
PA4 (LIST X (+ Y W2 DV H (- DS) AE))
PT1 (LIST X Y)
PT2 (LIST (+ X W2) (- Y W2))
PT3 (LIST (- X W2) (+ Y W2))
PT4 (LIST (- X WS2) (+ Y R S1))
PT5 (LIST (- X WS2) (- Y R S1))
PF1 (LIST X (+ Y W2 DV))
PF2 (LIST (- X W2 (- WF)) (+ Y W2 DV))
PF3 (LIST (- X W2) (+ Y W2 DV))
PF4 (LIST (- X W2) (+ Y W2 DV H2))
PF5 (LIST (- X W2 (- WF)) (+ Y W2 DV H2))
PF6 (LIST (- X R) (+ Y W2 DV H2))
PF7 (LIST (- X R) (+ Y W2 DV H))
PF8 (LIST (- X WS2) (+ Y W2 DV H))
PF9 (LIST (- X WS2) (+ Y W2 DV H (- DS)))
PF10 (LIST X (+ Y W2 DV H (- DS)))
PF11 (LIST X (+ Y W2 DV H2))
PF12 (LIST (+ X WS2) (+ Y W2 DV H (- DS)))
PF13 (LIST (+ X WS2) (+ Y W2 DV))
PF14 (LIST (+ X W2 (- S1)
            (+ Y W2 DV (/ H2 2.0))))
; збільшення області майбутнього креслення
; щоб усунути хибне спрацьовування прив'язок
(COMMAND "ZOOM" "W")
  (LIST (- X W2 AE AE) (- Y W2 AE AE))
  (LIST (+ X W2 AE AE) (+ Y W2 DV H AE AE)))
; перехід до шару осьових ліній
(COMMAND "LAYER" "SET" "OSI" "")
; креслення осей
(COMMAND "LINE" PA1 PA4 "")
(COMMAND "LINE" PA2 PA3 "")
; перехід до шару основних ліній
(COMMAND "LAYER" "SET" "OSN" "")
; креслення виду зверху
(COMMAND "RECTANG" "C" WF WF PT2 PT3)
(COMMAND "CIRCLE" "INT" PT1 R)
(COMMAND "CIRCLE" "INT" PT1 WS2)
(COMMAND "LINE" PT4 PT5 "")
(COMMAND "TRIM" "QUA" (LIST X (+ Y R)) "" PT4 PT5 "")
(SETQ SL (ENTLAST))
(COMMAND "MIRROR" SL ""
  (LIST X (+ Y R)) (LIST X (- Y R)) "N")
; креслення виду спереду
(COMMAND "PLINE" PF1 PF3 PF4 PF6 PF7 PF8 PF9 PF10 "")
(COMMAND "MIRROR"
  "NEA" PF2 ""

```

```

"INT" PF1 "INT" PF10
"N")
(COMMAND "LINE" "NEA" PF2 "NEA" PF5 "")
(COMMAND "LINE" "END" PF6 "NEA" PF11 "")
(COMMAND "LINE" "END" PF12 "NEA" PF13 "")
; перехід до шару штриховки
(COMMAND "LAYER" "SET" "HTCH" "")
(COMMAND "BHATCH" "P" "ANSI31" 1 0 PF14 "")
)
; збільшення існуючого креслення на весь екран
(COMMAND "ZOOM" "Extents")

```

Приклад 5. Розглянемо програму креслення болта з шестигранною голівкою класу точності В згідно ГОСТ 7798-70 (виконання 1). Ескізи болта (види спереду та збоку) наведені на рис. 5.9. Розміри болта для виконання 1 наведені у табл. 5.2, 5.3.

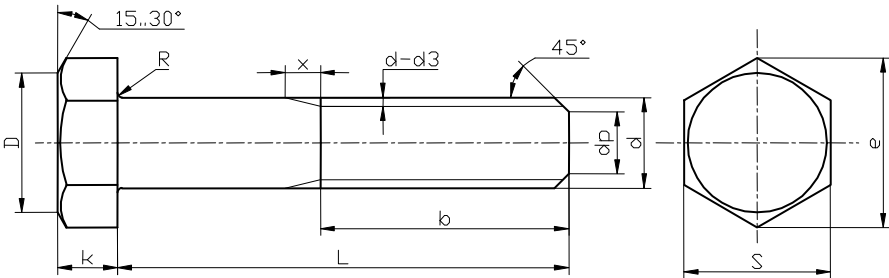


Рисунок 5.9 – Ескіз болта згідно ГОСТ 7798-70

Таблиця 5.2 – Розміри болта згідно ГОСТ 7798-70

Номинальний діаметр різьби, d	6	8	10	12	(14)	16	(18)	20	(22)	24	(27)	30	36	42	48
Крок різьби, P	крупний	1	1,25	1,5	1,75	2		2,5		3	3,5	4	4,5	5	
	мілкий	–	1	1,25		1,5		2		3		4		5	
Розмір «під ключ», S	10	13	16	18	21	24	27	30	34	36	41	46	55	65	75
Висота голівки, k	4,0	5,3	6,4	7,5	8,8	10,0	12,0	12,5	14,0	15,0	17,0	18,7	22,5	26,0	30,0
Діаметр описаного кола e , не менш	10,9	14,2	17,6	19,9	22,8	26,2	29,6	33,0	37,3	39,6	45,2	50,9	60,8	71,3	82,6

Продовження таблиці 5.2

Номинальний діаметр різьби, d	6	8	10	12	(14)	16	(18)	20	(22)	24	(27)	30	36	42	48
Радіус торця, dp	4,0	5,5	7,0	8,5	10,0	12,0	13,0	15,0	17,0	18,0	21,0	23,0	28,0	32,0	38,0
Радіус скруглення під голівкою, R	0,25	0,40	0,60			0,80			1,00			1,20	1,60		

Примітки:

1 Розміри болтів, наведені у дужках, використовувати не рекомендується.

2 Довжина нормального збігу різьби x дорівнює $2,5P$, короткого – $1,25P$, де P – крок різьби.

3 Глибина різьби $d-d_3 = 0,613434661P$, де P – крок різьби.

4 $D = 0.95S$

Таблиця 5.3 – Розміри болта згідно ГОСТ 7798-70

Довжина болта, L	Довжина різьби b при номінальному діаметрі різьби d														
	6	8	10	12	(14)	16	(18)	20	(22)	24	(27)	30	36	42	48
8	+	+	–	–	–	–	–	–	–	–	–	–	–	–	–
10	+	+	+	–	–	–	–	–	–	–	–	–	–	–	–
12	+	+	+	–	–	–	–	–	–	–	–	–	–	–	–
14	+	+	+	+	–	–	–	–	–	–	–	–	–	–	–
16	+	+	+	+	+	–	–	–	–	–	–	–	–	–	–
(18)	+	+	+	+	+	+	–	–	–	–	–	–	–	–	–
20	+	+	+	+	+	+	+	–	–	–	–	–	–	–	–
(22)	18	+	+	+	+	+	+	–	–	–	–	–	–	–	–
25	18	+	+	+	+	+	+	+	–	–	–	–	–	–	–
(28)	18	22	+	+	+	+	+	+	+	–	–	–	–	–	–
30	18	22	+	+	+	+	+	+	+	–	–	–	–	–	–
(32)	18	22	26	+	+	+	+	+	+	+	–	–	–	–	–
35	18	22	26	30	+	+	+	+	+	+	+	–	–	–	–
(38)	18	22	26	30	+	+	+	+	+	+	+	–	–	–	–
40	18	22	26	30	34	+	+	+	+	+	+	+	–	–	–
45	18	22	26	30	34	38	+	+	+	+	+	+	–	–	–
50	18	22	26	30	34	38	42	+	+	+	+	+	+	–	–

Продовження таблиці 5.3

Довжина болта, L	Довжина різьби b при номінальному діаметрі різьби d														
	6	8	10	12	(14)	16	(18)	20	(22)	24	(27)	30	36	42	48
55	18	22	26	30	34	38	42	46	+	+	+	+	+	+	—
60	18	22	26	30	34	38	42	46	50	+	+	+	+	+	—
65	18	22	26	30	34	38	42	46	50	54	+	+	+	+	—
70	18	22	26	30	34	38	42	46	50	54	60	+	+	+	+
75	18	22	26	30	34	38	42	46	50	54	60	66	+	+	+
80	18	22	26	30	34	38	42	46	50	54	60	66	+	+	+
(85)	18	22	26	30	34	38	42	46	50	54	60	66	+	+	+
90	18	22	26	30	34	38	42	46	50	54	60	66	78	+	+
(95)	—	22	26	30	34	38	42	46	50	54	60	66	78	+	+
100	—	22	26	30	34	38	42	46	50	54	60	66	78	+	+
(105)	—	—	26	30	34	38	42	46	50	54	60	66	78	90	+
110	—	—	26	30	34	38	42	46	50	54	60	66	78	90	+
(115)	—	—	26	30	34	38	42	46	50	54	60	66	78	90	102
120	—	—	26	30	34	38	42	46	50	54	60	66	78	90	102
(125)	—	—	26	30	34	38	42	46	50	54	60	66	78	90	102
130	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
140	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
150	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
160	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
170	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
180	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
190	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
200	—	—	32	36	40	44	48	52	56	60	66	72	84	96	108
220	—	—	—	49	57	57	61	65	69	73	79	85	97	109	121
240	—	—	—	49	57	57	61	65	69	73	79	85	97	109	121
260	—	—	—	49	57	57	61	65	69	73	79	85	97	109	121
280	—	—	—	—	57	57	61	65	69	73	79	85	97	109	121
300	—	—	—	—	57	57	61	65	69	73	79	85	97	109	121

Примітки:

1 Розміри болтів, наведені у дужках, використовувати не рекомендується.

2 Болти, для яких значення b розташовані над ламаною лінією, допускається виготовляти з довжиною різьби до голівки.

3 Знаком + відмічені болти з різьбою вздовж всього стрижня без збігу.

Згідно тексту ГОСТ 7798-70 деякі розміри болта визначені окремими стандартами (за таблицями або за співвідношеннями): різьба за ГОСТ 24705-81, збіги та недорізи різьби за ГОСТ 27148-86, кінці за ГОСТ 12414-94, радіуси під голівкою за ГОСТ 24670-81. Це є досить типовим, оскільки дозволяє скоротити текст стандарту. Вищевказані розміри є необхідними для виконання креслення, тому вони також позначені на рис. 5.9, а у табл. 5.1 знаходяться нижче подвійної лінії та у примітках. Розміри, що не відносяться до виконання 1 або дублюються, рекомендується не вносити до програми (наприклад, розмір $e = 2 \cdot S / \sqrt{3}$, якщо шестикутник правильний).

Для більшої наочності подальших розрахунків потрібно проаналізувати ці ескізи, розбити їх на мінімальну кількість примітивів та операцій, виявити потрібну множину прив'язок, накреслити ескіз розташування допоміжних точок (рис. 5.10) та обрати базову точку для кожного виду (точки PT1 та PT2). Товщина ліній не показана через велику кількість дрібних елементів на ескізі.

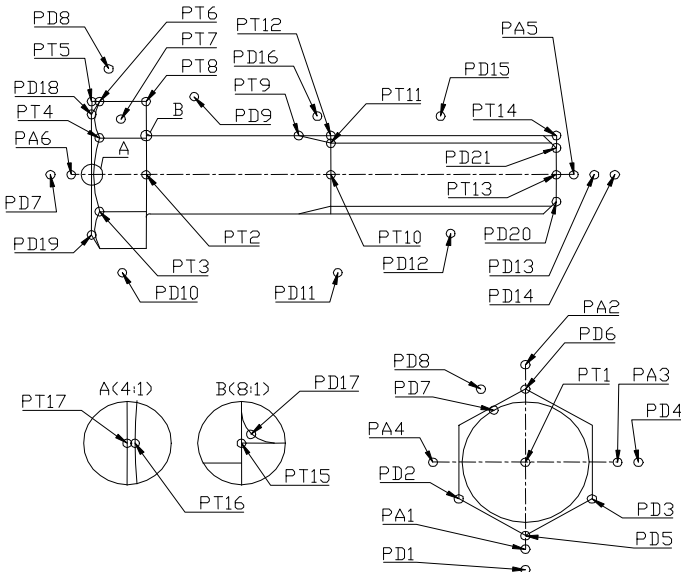


Рисунок 5.10 – Ескіз розташування допоміжних точок

Зважаючи на вид креслення, потрібно забезпечити програмне створення чотирьох шарів – для основних товстих ліній, для осевих тонких, для штриховки та для розмірів. Також дати можливість створення нових (або використання існуючих) шарів з іменами, вказаними користувачем.

Особливу увагу звернути на забезпечення безпомилкового введення даних.

Програма прикладу повністю функціональна, за виключенням відсутності повноцінної функції креслення головного виду. Базова функція **GOST7798-70** викликається без параметрів, тобто для виклику необхідно набрати (GOST7798-70) в командному рядку.

Приклад діалогового введення вхідних даних (у т.ч. обробки некоректних значень) та результат роботи програми представлені на рис. 5.11, 5.12.

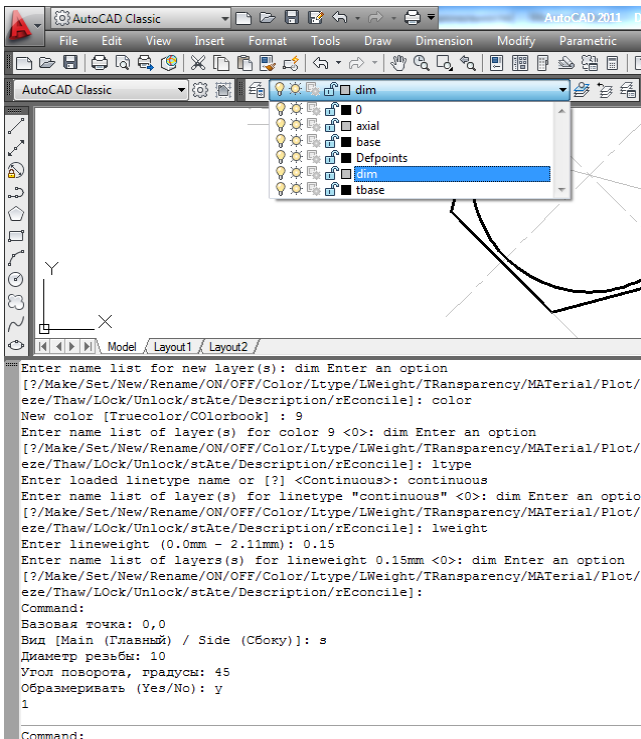


Рисунок 5.11 – Діалог з користувачем

Текст програми:

```

; рядкові константи інтерфейсу користувача
(setq ui_err "\nПовторите ввод!")
      ; список можливих відповідей (Yes/No)
      yn_list (list "y" "yes" "n" "no")
)
(defun user_input (/
  d_other      ; зв'язок d різби з іншими параметрами
  L_all        ; значення довжини болта
  L_unrec      ; nereкомендовані значення довжини болта
  d_Lb        ; (діаметр (мін_довж макс_довж довж_різби)...)
  d_all        ; значення діаметра болта
  d_unrec      ; nereкомендовані значення діаметра болта
  base_point   ; базова точка
  view         ; вид деталі
  d            ; діаметр болта
  views        ; список можливих значень видів
  ui_base      ; рядкові константи інтерфейсу користувача
  ui_view
  ui_dia
  ui_unrec
  ui_len
  ui_ang
  ui_dim
  ui_rout
  L_d          ; список значень L для введенного d
  L_ds         ; рядкове подання списку L_d
  a_rot        ; кут обертання
  putdim       ; режим простановки розмірів
  req_par      ; обов'язкові параметри (для всіх видів)
  aux_par      ; додаткові параметри (для головного виду)
  Li           ; допоміжна змінна для організації циклу
  lims         ; список списків
              ; (зв'язок b з діапазоном значень L)
  other        ; список параметрів залежних тільки від d
)
      ; d S Pb Ps k dp R
(setq d_other (list
  '( 6.0 10.0 1.00 nil 4.0 4.0 0.25)
  '( 8.0 13.0 1.25 1.00 5.3 5.5 0.40)
  '(10.0 16.0 1.50 1.25 6.4 7.0 0.40)
  '(12.0 18.0 1.75 1.25 7.5 8.5 0.60)
  '(14.0 21.0 2.00 1.50 8.8 10.0 0.60)
  '(16.0 24.0 2.00 1.50 10.0 12.0 0.60)
  '(18.0 27.0 2.50 1.50 12.0 13.0 0.60)
  '(20.0 30.0 2.50 1.50 12.5 15.0 0.80)
  '(22.0 34.0 2.50 1.50 14.0 17.0 0.80)

```

```

' (24.0 36.0 3.00 2.00 15.0 18.0 0.80)
' (27.0 41.0 3.00 2.00 17.0 21.0 1.00)
' (30.0 46.0 3.50 2.00 18.7 23.0 1.00)
' (36.0 55.0 4.00 3.00 22.5 28.0 1.00)
' (42.0 65.0 4.50 3.00 26.0 32.0 1.20)
' (48.0 75.0 5.00 3.00 30.0 38.0 1.60))
L_all (list
  8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0
  25.0 28.0 30.0 32.0 35.0 38.0 40.0 45.0
  50.0 55.0 60.0 65.0 70.0 75.0 80.0 85.0
  90.0 95.0 100.0 105.0 110.0 115.0 120.0 125.0
  130.0 140.0 150.0 160.0 170.0 180.0 190.0 200.0
  220.0 240.0 260.0 280.0 300.0)
L_unrec (list
  18.0 22.0 28.0 32.0 38.0
  85.0 95.0 105.0 115.0 125.0)
; якщо довжина різьби = 0.0
; то довжина різьби = L, без збігу
d_Lb (list
' ( 6.0 ( 8.0 20.0 0.0) ( 22.0 90.0 18.0))
' ( 8.0 ( 8.0 25.0 0.0) ( 28.0 100.0 22.0))
' (10.0 ( 10.0 30.0 0.0) ( 32.0 125.0 26.0)
  (130.0 200.0 32.0))
' (12.0 ( 14.0 32.0 0.0) ( 35.0 125.0 30.0)
  (130.0 200.0 36.0) (220.0 260.0 49.0))
' (14.0 ( 16.0 38.0 0.0) ( 40.0 125.0 34.0)
  (130.0 200.0 40.0) (220.0 300.0 57.0))
' (16.0 ( 18.0 40.0 0.0) ( 45.0 125.0 38.0)
  (130.0 200.0 44.0) (220.0 300.0 57.0))
' (18.0 ( 20.0 45.0 0.0) ( 50.0 125.0 42.0)
  (130.0 200.0 48.0) (220.0 300.0 61.0))
' (20.0 ( 25.0 50.0 0.0) ( 55.0 125.0 46.0)
  (130.0 200.0 52.0) (220.0 300.0 65.0))
' (22.0 ( 28.0 55.0 0.0) ( 60.0 125.0 50.0)
  (130.0 200.0 56.0) (220.0 300.0 69.0))
' (24.0 ( 32.0 60.0 0.0) ( 65.0 125.0 54.0)
  (130.0 200.0 60.0) (220.0 300.0 73.0))
' (27.0 ( 35.0 65.0 0.0) ( 70.0 125.0 60.0)
  (130.0 200.0 66.0) (220.0 300.0 79.0))
' (30.0 ( 40.0 70.0 0.0) ( 75.0 125.0 66.0)
  (130.0 200.0 72.0) (220.0 300.0 85.0))
' (36.0 ( 50.0 85.0 0.0) ( 90.0 125.0 78.0)
  (130.0 200.0 84.0) (220.0 300.0 97.0))
' (42.0 ( 55.0 100.0 0.0) (105.0 125.0 90.0)
  (130.0 200.0 96.0) (220.0 300.0 109.0))
' (48.0 ( 70.0 110.0 0.0) (115.0 125.0 102.0)
  (130.0 200.0 108.0) (220.0 300.0 121.0))

```

```

)
d_all (list
  6.0 8.0 10.0 12.0 14.0 16.0 18.0 20.0 22.0
  24.0 27.0 30.0 36.0 42.0 48.0)
d_unrec (list 14.0 18.0 22.0 27.0)
base_point (list 0.0 0.0)
view "M"
d (caar d_Lb)
views (list "m" "main" "s" "side")
ui_base "\nБазовая точка: "
ui_view "\nВид [Main (Главный) / Side (Сбоку)]: "
ui_dia "\nДиаметр резьбы: "
ui_unrec "\nЗначение не рекомендуется!"
ui_len "\nДлина болта"
ui_ang "\nУгол поворота, градусы: "
ui_dim "\nОбразмеривать (Yes/No): "
ui_rout "\nCбег резьбы [Normal (Нормальный)/ Short
(Короткий)]: "
  L_d nil
  L_ds ""
  a_rot 0.0
  putdim "y"
  req_par nil
  aux_par nil
  runouts (list "n" "normal" "s" "short")
  runout "n"
)
;-----
; базова точка
(setq base_point (getpoint ui_base))
; вид (з приведениям до нижнього регістру)
(setq view (strcase (getstring ui_view) T))
(while (= (member view views) nil)
  (prompt ui_err)
  (setq view (strcase (getstring ui_view) T))
)
; діаметр різьби
(setq d (getreal ui_dia))
(while (= (member d d_all) nil)
  (prompt ui_err)
  (setq d (getreal ui_dia))
)
(if (/= (member d d_unrec) nil)
  (prompt ui_unrec)
)
; додаткові параметри (для головного виду)
; довжина болта

```

```

(if (or (= view "m") (= view "main"))
  (progn
    (setq lims (cdr (assoc d d_Lb)))
    ; створення списку та рядкового подання
    ; дозволених значень L для введенного значення d
    (foreach Li L_all
      (if (and (>= Li (caar lims))
              (<= Li (cadr (last lims))))
        (progn
          (setq L_d (append L_d (list Li)))
          (setq L_ds (strcat L_ds (rtos Li 2 2) ",")))
        )
      )
    )
  )
; довжина різьби (пропонує список дозволених значень)
(setq L (getreal
  (strcat ui_len
    "<"
    (substr L_ds 1 (- (strlen L_ds) 1))
    ">: ")
))
(while (= (member L L_d) nil)
  (prompt ui_err)
  (setq L (getreal (strcat ui_len ": ")))
)
(if (/= (member L L_unrec) nil)
  (prompt ui_unrec)
)
; обчислення b з умов, визначених списком d_Lb
(foreach Li lims
  (if (and (>= L (car Li)) (<= L (cadr Li)))
    (setq b (caddr Li))
  )
)
; обробка випадку відсутності збігу (b=0)
(if (zerop b)
  (setq b L)
)
; запит на нормальний чи короткий збір
(setq runout (strcase (getstring ui_rout) T))
(while (= (member runout runouts) nil)
  (prompt ui_err)
  (setq runout (strcase (getstring ui_rout) T))
)
)
)
; список додаткових параметрів головного виду
(setq other (cdr (assoc d d_other)))

```

```

; кут обертання
(setq a_rot (getreal ui_ang))
(while (not (numberp a_rot))
  (prompt ui_err)
  (setq a_rot (getreal ui_ang))
)
; простановка розмірів
(setq putdim (strcase (getstring ui_dim) T))
(while (= (member putdim yn_list) nil)
  (prompt ui_err)
  (setq putdim (strcase (getstring ui_dim) T))
)
; повернути введені опції та значення списком виду
; req_par
; base-x base-y view S a_rot putdim
; aux_par
; d L b Pb Ps k dp R runout
(setq req_par (list (car base_point)
  (cadr base_point)
  view
  (car other)
  a_rot
  putdim))
(if (or (= view "m") (= view "main"))
  (setq req_par (append req_par
    (list d L b)
    (cdr other)
    (list runout))
  )
)
; append повертає список введених даних
; у випадку виду збоку (останній обчислений вираз)
(append req_par))
(defun olayer (lcol ltyp lwei ui_str def_lname)
  (setq lname (getstring (strcat ui_str
    " <"
    def_lname
    ">: ")))
)
(if (= lname "")
  (setq lname def_lname)
)
(command "layer" "new" lname
  "color" lcol lname
  "ltype" ltyp lname
  "lweight" lwei lname "")
(eval lname)

```

```

)
(defun tune_env (/ ui_lb ui_lax ui_ltb ui_ldm)
  (command "snap" "off")
  (command "grid" "off")
  (setvar "osmode" 0)
  (setvar "dimto" 1)
  (setvar "dimexo" 0)
  (setvar "aperture" 5)
  (setvar "ltscale" 2)
  (setq ui_lb "\nВведите имя слоя основных линий"
        ui_lax "\nВведите имя слоя осевых линий"
        ui_ltb "\nВведите имя слоя тонких линий"
        ui_ldm "\nВведите имя слоя размерных линий")
  (setq layer_base
        (olayer "7" "continuous" "0.30" ui_lb "base")
        layer_axial
        (olayer "9" "center" "0.15" ui_lax "axial")
        layer_thin
        (olayer "7" "continuous" "0.15" ui_ltb "tbase")
        layer_dims
        (olayer "9" "continuous" "0.15" ui_ldm "dim"))
)
(defun plot_side (pd / s2 e2 d_big
                  pt1
                  pa1 pa2 pa3 pa4
                  pd1 pd2 pd3 pd4 pd5 pd5 pd6 pd7 pd8
                  pz1 pz2)
  (setq pt1 (list (car pd) (cadr pd)))
  s2 (/ (nth 3 pd) 2.0)
  e2 (/ (nth 3 pd) (sqrt 3))
  pa1 (list (car pd) (- (cadr pd) e2 5.0))
  pa2 (list (car pd) (+ (cadr pd) e2 5.0))
  pa3 (list (+ (car pd) s2 5.0) (cadr pd))
  pa4 (list (- (car pd) s2 5.0) (cadr pd))
  pd1 (list (car pd) (- (cadr pd) e2 10.0))
  pd2 (polar pt1 (/ (* pi 7.0) 6.0) e2)
  pd3 (polar pt1 (/ (- pi) 6.0) e2)
  pd4 (list (+ (car pd) s2 10.0) (cadr pd))
  pd5 (list (car pd) (- (cadr pd) e2))
  pd6 (list (car pd) (+ (cadr pd) e2))
  d_big (* (nth 3 pd) 0.95)
  pd7 (polar pt1 (/ (* pi 2.0) 3.0) (* d_big 0.5))
  pd8 (polar pt1 (/ (* pi 2.0) 3.0) (* d_big 0.7))
  pz1 (list (- (car pd) s2 10.0) (- (cadr pd) e2 15.0))
  pz2 (list (+ (car pd) s2 10.0) (+ (cadr pd) e2 15.0))
)
(setq cside (ssadd))

```

```

(command "zoom" "window" pz1 pz2)
(command "layer" "set" layer_axial "")
(command "line" pa1 pa2 "")
(setq cside (ssadd (entlast) cside))
(command "line" pa3 pa4 "")
(setq cside (ssadd (entlast) cside))
(command "layer" "set" layer_base "")
(command "circle" pt1 (/ d big 2.0))
(setq cside (ssadd (entlast) cside))
(command "polygon" 6 pt1 "C" s2)
(setq cside (ssadd (entlast) cside))
(command "rotate" (entlast) "" pt1 30)
(if (= (substr (nth 5 pd) 1 1) "y")
    (progn
      (command "layer" "set" layer_dims "")
      (command "dimlinear" "end" pd2 "end" pd3 pd1)
      (setq cside (ssadd (entlast) cside))
      (command "dimlinear" "end" pd5 "end" pd6 pd4)
      (setq cside (ssadd (entlast) cside))
      (command "zoom" "Extents")
      (command "dimdiameter" "nea" pd7 pd8)
      (setq cside (ssadd (entlast) cside))
    )
  )
(if (/= (nth 4 pd) 0.0)
    (command "rotate" cside "" pt1 (nth 4 pd)))
)
(defun plot_main (pd)
  (print "Функція відсутня")
)
(defun GOST7798-70 ()
  (tune_env)
  ; вимкнення виводу повідомлень під час виконання команд
  (setvar "cmdecho" 0)
  (setq plot_data (user_input))
  (if (= (substr (nth 2 plot_data) 1 1) "s")
      ; вид збоку
      (plot_side plot_data)
      ; головний вид
      (plot_main plot_data)
  )
  (setvar "cmdecho" 1)
  (setvar "lwdisplay" 1)
)

```

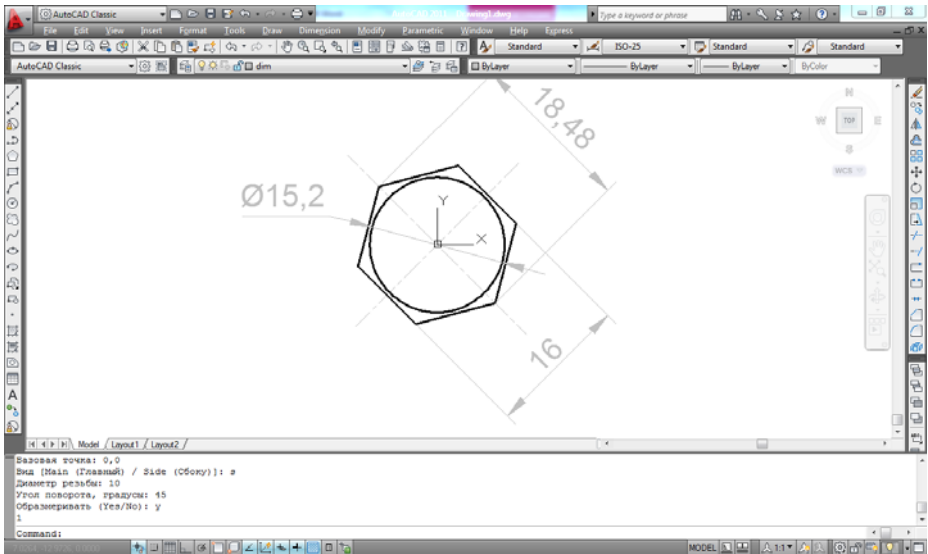


Рисунок 5.12 – Результати роботи програми для прикладу 5

У програмі були використані наступні нові команди AutoCAD (порівняно з попередніми прикладами):

POLYGON – креслення правильного рівностороннього замкненого багатокутника з даною кількістю сторін (від 3 до 1024), за центром та радіусом вписаного/описаного кола або за двома точками однієї сторони;

ROTATE – обертання вказаних примітивів (або іменованого набору) навколо вказаної точки на заданий кут (можливе переміщення або копіювання примітивів);

DIMLINEAR – постановка лінійного розміру між двома вказаними точками;

DIMDIAMETER – постановка діаметрального розміру на вказаному колі або дузі;

У програмі були використані специфічні функції AutoLISP:

(CAR list) – повертає перший елемент (голову) списку;

(CADR list) – повертає другий елемент списку;

(CADDR list) – повертає третій елемент списку;

Функції **CAR**, **CADR**, **CADDR** часто використовуються для доступу до X, Y, Z координат точок відповідно.

(CDR list) – повертає список всіх елементів крім першого (хвіст списку);

(CAAR list) – повертає голову другого елемента списку, еквівалентний до виразу **(CAR (CAR list))**;

(APPEND [list...]) – повертає конкатенацію вказаних списків;

(ASSOC element alist) – повертає результат асоціативного пошуку елемента *element* (список) у асоціативному списку списків *alist*;

(MEMBER expr lst) – виконує пошук виразу *expr* у списку *lst*; повертає остаток списку *lst* (включно з першим входженням виразу); якщо вираз не знайдено – повертає *nil*, таким чином можна виконувати перевірку на входження елементу в список;

(NTH n list) – повертає *n*-ний елемент списку, $n \geq 0$;

(SSADD [ename [ss]]) – використовується для створення нового набору примітивів (якщо викликається без аргументів), або для додавання примітиву *ename* до набору *ss*; повертає ім'я модифікованого набору;

(GETPOINT [pt] [msg]) – повертає точку (список), координати якої або введені в командному рядку, або вказані мишею; параметр *pt* – це точка (або не рядковий параметр), між нею та курсором буде динамічно проводитися лінія; параметр *msg* – це рядок тексту запрошення у командному рядку;

(GETSTRING [cr] [msg]) – повертає рядок, введений в командному рядку; якщо параметр *cr* дорівнює *T*, то в рядку можна вводити пробіли, тоді завершати ввід потрібно натисненням ENTER; параметр *msg* – це рядок тексту запрошення у командному рядку;

(GETREAL [msg]) – повертає дійсне число, введене у командному рядку; параметр *msg* – це рядок тексту запрошення у командному рядку;

У відповідь на запити функцій **GET*** не допускається вводити вирази AutoLISP.

(EVAL expr) – повертає результат обчислення виразу *expr*; може використовуватися для повертання потрібного значення у функціях користувача (якщо воно не є останнім обчисленням);

(ZEROP number) – перевірка чи дорівнює нулю число *number*;

(NUMBERP item) – перевірка чи є аргумент *item* числом;

(RTOS number [mode [precision]]) – повертає рядкове подання числа **number**; якщо воно дійсне, то необхідно вказати формат перетворення – аргумент **mode**:

- 1 – науковий;
- 2 – десятковий;
- 3 – інженерний (фути та десяткові дюйми);
- 4 – архітектурний (фути та дробові дюйми);
- 5 – дробовий;

аргумент **precision** встановлює кількість знаків після десяткової крапки;

(STRCASE string [which]) – перетворення регістру символів рядку **string**, якщо аргумент **which** дорівнює **T**, то регістр нижній, інакше – верхній;

(STRCAT [string [string]...]) – повертає результат конкатенації рядкових аргументів **string**;

(STRLEN [string]...) – повертає кількість символів у рядку **string** (або суму кількостей для всіх вказаних рядків, або 0, якщо викликати без аргументів);

(SUBSTR string start [length]) – повертає підрядок рядка **string**, починаючи з символу з номером **start** (нумерація з 1), довжиною **length**, якщо вона не вказана – то до кінця рядку;

(WHILE testexpr [expr...]) – обчислює вирази **expr** доти, доки **testexpr** дорівнюватиме **nil**;

(FOREACH name list [expr...]) – обчислює вирази **expr** для кожного елемента **name** списку **list**;

(POLAR pt ang dist) – повертає координати точки, яка знаходиться на відстані **dist** та під кутом **ang** від точки **pt** (кут виражається у радіанах та відраховується від напрямку вісі X проти годинникової стрілки);

5.4 Методика створення сервісних бібліотек в САПР AutoCAD

Примітка. При виконанні курсового проекту рекомендується використання власних імен файлів слайдів, бібліотек, шляхів розташування файлів та інших елементів з урахуванням зміни усіх цих імен, за умов дотримання контексту даних методичних вказівок.

5.4.1 Розробка LISP-програм та створення слайдів

Перед початком роботи потрібно створити робочу папку. Далі необхідно додати створену папку до списку шляхів пошуку шрифтів, програм, бібліотек, файлів меню. Для цього у пункті головного меню **Tools** обрати підпункт **Options...** (або **Options...** в контекстному меню), відкриється вікно **Options**, в якому перейти на закладку **Files**, відкрити розділ **Support File Search Path...**, натиснути кнопку **Add...** та або вписати відомий шлях до робочої папки, або натиснути кнопку **Browse...** та знайти робочу папку вручну (див. рис. 5.13).

Розробити LISP-програму креслення заданого стандартного кріпильного елемента. Зберегти програмний файл (*.lsp) в робочу папку. Створена програма повинна забезпечувати наступне:

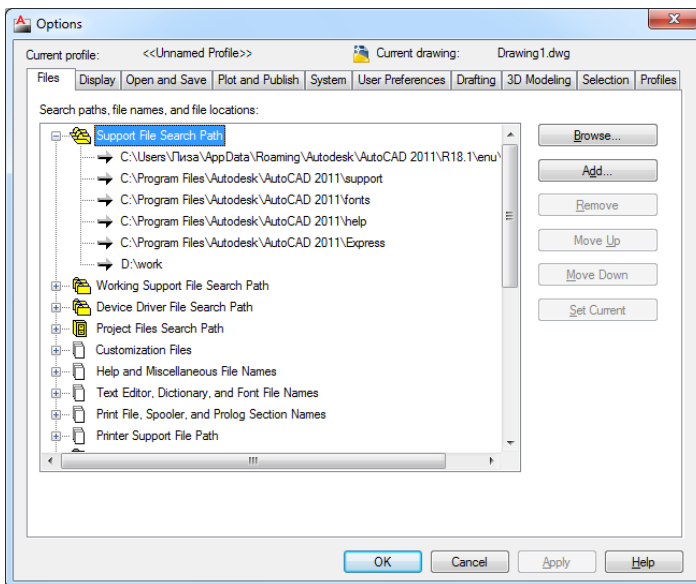


Рисунок 5.13 – Додавання шляху до робочої папки

- діалог з користувачем, у т.ч. контроль вхідних даних;
- креслення головного виду деталі та виду зверху (збоку);
- розташування елемента в необхідному місці креслення;
- розташування елемента під необхідним кутом;
- можливість постановки розмірів на кресленні.

Інтеграція програми до середовища AutoCAD відбувається за рахунок додавання спадаючого пункту головного меню, після звернення до якого повинне відкриватись **мозаїчне меню**, що містить **слайди** кріпильних елементів.

Для створення слайду необхідно програмно виконати креслення стандартного елемента. Зберігати його немає необхідності. Потім, у командному рядку AutoCAD ввести команду MSLIDE для створення файлу слайда, який має розширення *.sld. Зверніть увагу, що слайд розміром 70х46 пікселів буде відображатися в меню так, як креслення відображається на екрані у мить створення слайду, тому рекомендується не перевантажувати креслення дрібними елементами та написами.

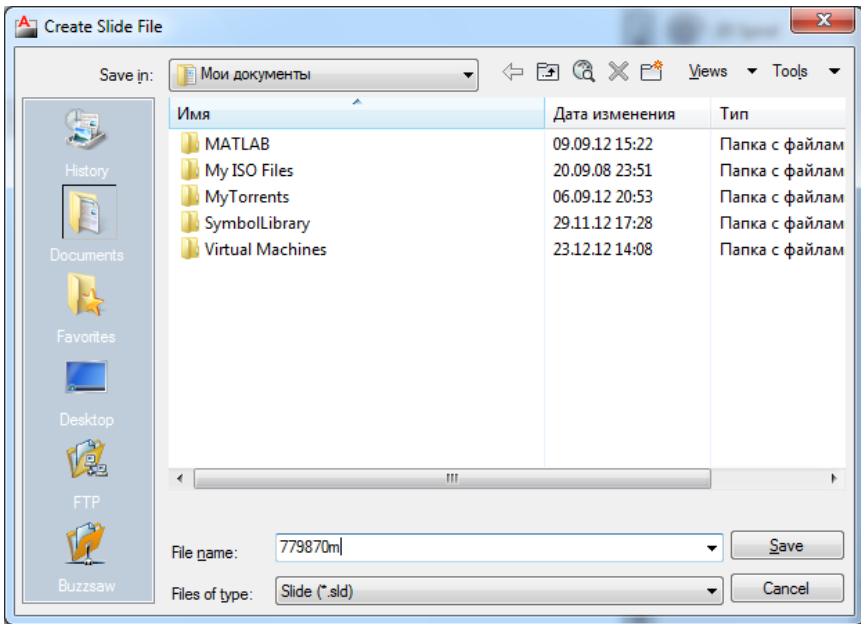


Рисунок 5.14 – Збереження файлу слайду

На рис. 5.14 відображено вікно з запитом на збереження файлу. При цьому, необхідно ввести ім'я слайду та записати його в робочу папку. У даному випадку створено файл з ім'ям 779870m.sld. Після створення потрібної кількості слайдів їх необхідно об'єднати в одну

або декілька бібліотек слайдів (розбиваючи за функціональним призначенням).

5.4.2 Створення файлу бібліотеки слайдів

Після того, як потрібні слайди створені, необхідно об'єднати їх у файл бібліотеки слайдів, який має розширення **.slb*.

Спочатку створимо файл списку усіх слайдів з ім'ям, наприклад, *list.txt* у будь-якому текстовому редакторі з переліком усіх імен слайдів (див. рис. 5.15). Збережемо цей файл у робочу папку.

Далі, необхідно скопіювати утиліту *slidelib.exe* у робочу папку. Вона знаходиться у папці, в яку встановлено AutoCAD (для версії 2006 та всі наступних), або у підпапці *Support* (для попередніх версій).

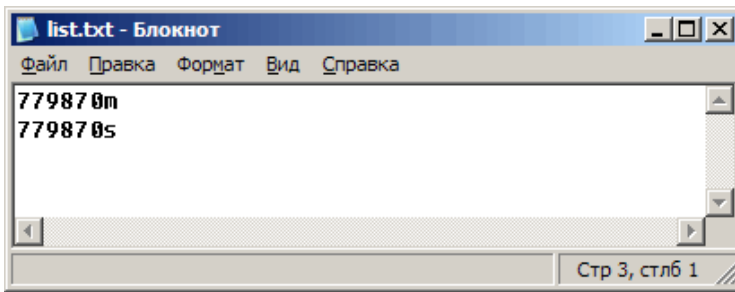


Рисунок 5.15 – Файл-список слайдів

За допомогою цієї стандартної утиліти створимо файл бібліотеки слайдів з ім'ям *lib7798.slb*. Виклик здійснюється наступним чином з командного рядка (необхідно перейти в робочу папку з текстовим файлом *list.txt* та слайдами створеними раніше):

```
SLIDELIB.EXE LIB7798<LIST.TXT
```

5.4.3 Створення файлу меню сервісної бібліотеки AutoCAD

У будь-якому текстовому редакторі створимо файл меню сервісної бібліотеки з ім'ям *acad_779870.mni*. Наведемо опис змісту файлу на рисунку 5.16.

```
***MENUGROUP=Screw
```

Команда створення групи меню *Screw*; група створюється для того, щоб при завантаженні файлу меню нове визначення елементів інтерфейсу не конфліктувало з існуючим; якщо група не буде

визначена, то завантаження файлу меню користувача викликає вивантаження основного файлу меню.

***pop1

Команда створення кнопки меню pop1

ID_MnScrew [Бібліотека]

Ідентифікатор та назва кнопки меню.

ID_ScrewImages [Болты...]

\$I=Screw.imglibrary \$I=Screw.*

Ідентифікатор, назва пункту меню та пов'язаний з ним макрос, який виконує заміну елемента інтерфейсу і має вид:

\$section=infogroup.menuname,

де ***\$*** – завантажує елемент;

section – визначає тип елемента (A1–A4 для меню Aux від 1 до 4, B1–B4 для кнопок миші від 1 до 4, P0–P16 для спадаючих меню від 0 до 16, I для мозаїчного меню, S для екранного меню, T1–T4 для меню планшета від 1 до 4);

infogroup – визначає, що меню ***menuname*** належить до групи меню ***infogroup***;

menuname – визначає ім'я розділу меню, що вставляється;

Для відображення заміненого елемента слугує друга частина макросу \$I=Screw.*;

***IMAGE

Початок розділу, який містить визначення мозаїчних меню.

**imglibrary

Нове мозаїчне меню.

[Бібліотека]

Назва заголовку вікна мозаїчного меню.

[lib7798 (779870m, Болт ГОСТ 7798–70)]

Опис слайда в мозаїчному меню, має наступний формат:

[lib_name(sld_name,sld_desc)]

де ***lib_name*** – ім'я файлу бібліотеки слайдів;

sld_name – ім'я слайду в бібліотеці;

sld_desc – текст назви слайду в мозаїчному меню.

```
^C^C(load "test5.LSP") (GOST7798-70)
```

Макрос, пов'язаний зі слайдом, виконується при подвійному натисканні на слайд.

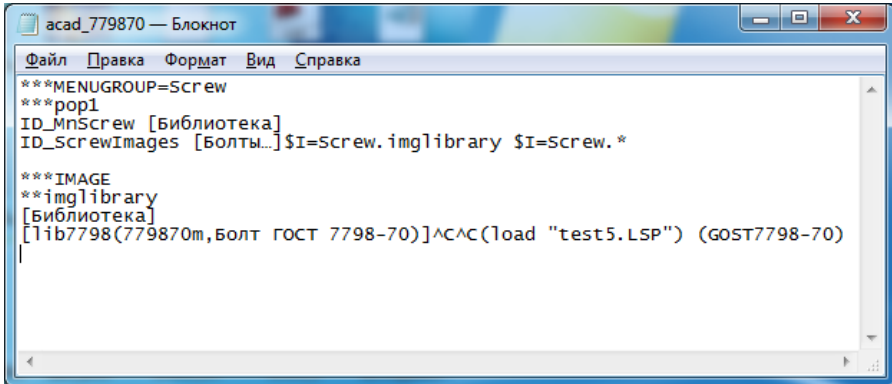


Рисунок 5.16 – Зміст файлу меню сервісної бібліотеки.

Макрос визначає послідовність операцій, яка виконується після активації елементу інтерфейсу. Може містити команди, спеціальні символи, вирази DIESEL (Direct Interpretively Evaluated String Expression Language – Безпосередньо Інтерпретативно Обчислювана Мова Рядкових Виразів) або функції AutoLISP.

Довжина макросу практично необмежена, також він може містити спеціальні символи, наведені у табл. 5.4.

Символ ^ еквівалентний натисненню клавіші CTRL, його можна комбінувати з іншими символами для утворення макросів.

Перед виконанням макросу слід пересвідчитися, що не виконується ніяка команда. Для автоматичної відміни команд на початку макросу слід вставити символи ^C^C. Для більшості команд достатньо писати ^C, але подвійний запис потрібен для відміни виконання команд постановки розмірів. Кожний символ в макросі є значимим, навіть пробіл.

Таблиця 5.4 – Спеціальні символи у макросах

Символ	Опис
;	Натиснення ENTER
^M	Натиснення ENTER
^I	Натиснення TAB
[пробіл]	Натиснення пробілу
\	Пауза для вводу користувача
_	Переклад команд локалізованих версій AutoCAD
=*	Відображення поточних меню (спадаючих верхнього рівня, мозаїчних)
*^C^C	Повтор команди до вводу іншої команди
\$	Вставка умовного макровиразу DIESEL (\$M=)
^B	Перемикання режимів прив'язки за сіткою (еквівалент CTRL+B)
^C	Відміна команди (еквівалент ESC)
^D	Перемикання динамічного відображення координат курсору (еквівалент CTRL+D)
^E	Установка наступної ізометричної площини (еквівалент CTRL+E)
^G	Перемикання сітки (еквівалент CTRL+G)
^H	Натиснення BACKSPACE
^O	Перемикання режиму ортогонального креслення
^P	Перемикання режимів MENCCHO
^Q	Дублювання всіх запитів, повідомлень командного рядку, виводу на принтер
^T	Перемикання планшета (еквівалент CTRL+T)
^V	Зміна поточного видового вікна
^Z	Нульовий символ, який запобігає автоматичному додаванню пробілу в кінці команди

5.4.4 Підключення файлу меню до середовища AutoCAD

Для підключення створеного файлу меню сервісної бібліотеки до середовища AutoCAD необхідно ввести команду **MENULOAD** у командному рядку, відкриється діалогове вікно **Load/Unload Customizations**, натиснути кнопку **Browse...** та вказати на

створений файл меню *acad_779870.mnu*, потім натиснути кнопку **Load**. Середовище AutoCAD прийме вигляд, наведений на рис. 5.17.

Після виклику пункту меню "Бібліотека" отримаємо вигляд робочого вікна сервісного додатку, показаний на рис. 5.18.

Можливим є використання декількох бібліотек за умов аналогічного підключення та використання унікальних ідентифікаторів елементів інтерфейсу.

Програма має бути написана таким чином, щоб її можна було легко модифікувати (модульна структура). Дані мають бути відокремлені від програми (наскільки це можливо зробити засобами AutoLISP), тобто виділені в окремі списки. Також всі повідомлення, які видає програма мають бути відокремлені від програмного коду і міститися в спеціальних змінних. Такий підхід до організації структури програми дозволяє з меншими витратами часу переписати її для креслення іншого стандартного елемента або використовувати готові частини даної програми при розробці інших сервісних додатків.

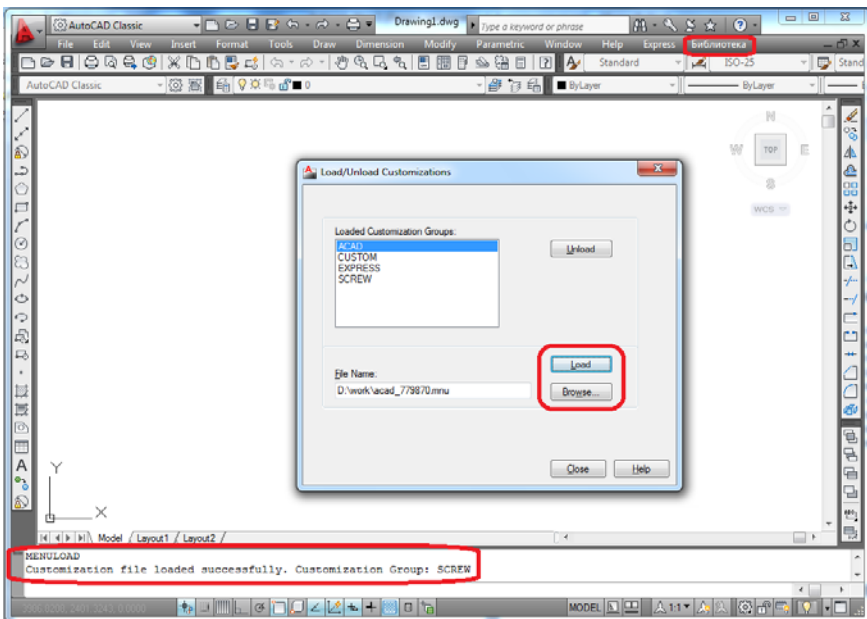


Рисунок 5.17 – Вікно завантаження файлів меню

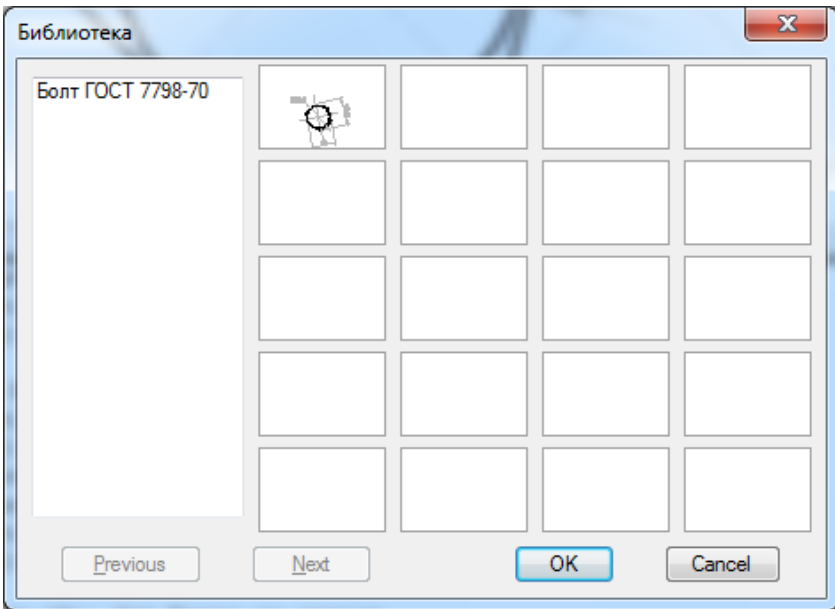


Рисунок 5.18 – Мозаїчне меню сервісного додатку

5.4.5 Налаштування інтерфейсу користувача AutoCAD 2011

Не зважаючи на те, що базова методика налаштування залишилася незмінною, порівняно з попередніми версіями AutoCAD, змінилося саме середовище, у якому налагоджується система.

Всі попередні опції налаштування лишаються доступними: створення, редагування та видалення елементів інтерфейсу, створення часткових файлів налаштування, використання макросів та розширених елементів (виразів DIESEL та підпрограм AutoLISP).

Однак, тепер налаштування здійснюється не ручним створенням або редагуванням текстових файлів MNU та MNS. Всі зміни виконуються через програмний інтерфейс, у діалоговому вікні ***Customize User Interface***.

У релізах AutoCAD до 2006 налаштування інтерфейсу користувача проводилося за рахунок редагування MNU та MNS файлів у текстовому редакторі. Ручне введення та перевірка даних

були рутинними операціями, які могли викликати помилки. В результаті навіть проста синтаксична помилка (наприклад, незакриті дужки) могла призвести до повної непридатності файлу меню, що змушувало вдаватися до ручного пошуку помилки.

Діалогове вікно ***Customize User Interface*** дозволяє користувачу візуалізувати процес налагодження, переміщення, додавання, видалення, модифікації елементів інтерфейсу. Діалогове вікно відображає властивості елементів та списки доступних опцій. Це значно зменшує можливість внесення помилки вручну.

Файли MNU та MNS були замінені одним XML-типом файлів – CUIx. Формат CUIx являється результатом розвитку формату файлів АПІ (CUI), який вперше з'явився в AutoCAD 2006.

Той факт, що файли CUIx мають формат XML дозволяє відстежувати процес налагодження системи. Під час оновлення версії програми всі настройки користувача автоматично інтегруються до нової установки. Формат XML також підтримує зворотну сумісність файлів налагодження. Це означає, що користувач може переглянути файл CUIx зі старшої версії у молодшій версії системи без знищення даних налагодження старшої версії. Однак, модифікація файлів CUIx старших версій у молодших версіях неможлива.

Відповідність старих та нових файлів меню наступна:

- MNU (Текстовий файл. У попередніх версіях визначав більшість елементів інтерфейсу користувача. Головний файл меню *acad.mnu* автоматично завантажувався після запуску системи);

- MNS (джерело файлу меню; ідентичний до MNU файлу, але не містить коментарів та форматування);

- MNC (скопійований текстовий файл; містить рядки команд та синтаксис, який визначає функціональність та вигляд елементів інтерфейсу) замінюються одним файлом CUIx (XML файл, який визначає більшість елементів інтерфейсу користувача. Головний CUI-файл *acad.cuix* автоматично завантажується після запуску системи).

Файли MNL (LISP файл меню; містить вирази AutoLISP, які використовуються елементами інтерфейсу) та MNR (файл ресурсів меню; містить зображення іконок елементів інтерфейсу) залишаються без змін.

AutoCAD 2011 зберігає сумісність зі старим типом файлів меню за рахунок їх перетворення до нового формату безпосередньо при завантаженні командою `MENULOAD`, що дозволяє використовувати

методику підключення сервісних додатків, описану в п. 5.4.4. Але нові можливості діалогу **Customize User Interface** дозволяють здійснити підключення сервісного додатку з нуля.

5.4.6 Підключення сервісного додатку до AutoCAD 2011

Діалогове вікно **Customize User Interface** викликається командою CUI, або необхідно в головному меню обрати пункт **Tools > Customize > Interface...** (див. рис. 5.19) Воно складається з двох закладок:

- **Customize** (Налагодження);
- **Transfer** (Передача).

Закладка **Customize** містить три розділи:

- **Customization in All File** (Налагодження в усіх файлах);
- **Command List** (Список Команд);
- **Properties** (Властивості).

Закладка **Transfer** містить два розділи:

- **Customizations in Main File** (Налагодження в головному файлі);
- **Customizations in New File** (Налагодження в новому файлі).

Необхідно перейти на закладку **Customize**, у розділ **Customization in All CUI Files** та у спадаючому меню обрати **Main Customization File (acad.cuix)** замість **All Customization Files** (див. рис. 5.20). Саме у деревовидній структурі файлу *acad.cuix* буде проводитись додавання елементів користувача (пункту меню та мозаїчного меню).

Далі треба виділити у структурі меню **ACAD** гілку **Menus** та у її контекстному меню обрати пункти **New > Menu** (див. рис. 5.21).

Буде створено новий пункт меню, для якого необхідно ввести назву (наприклад, **Кренезж**). Далі перейти до розділу **Command List** та натиснути кнопку **Create a new command** , щоб створити нову команду, яка буде вставлена як пункт меню **Кренезж**. Властивості нової команди встановлюються у розділі **Properties** (ім'я, опис, асоційований макрос, ідентифікатор елемента, іконка) (див. рис. 5.22). Введіть назву команди **Болты**, макрос:

```
$I=ACAD.imglibrary $I=ACAD.*.
```

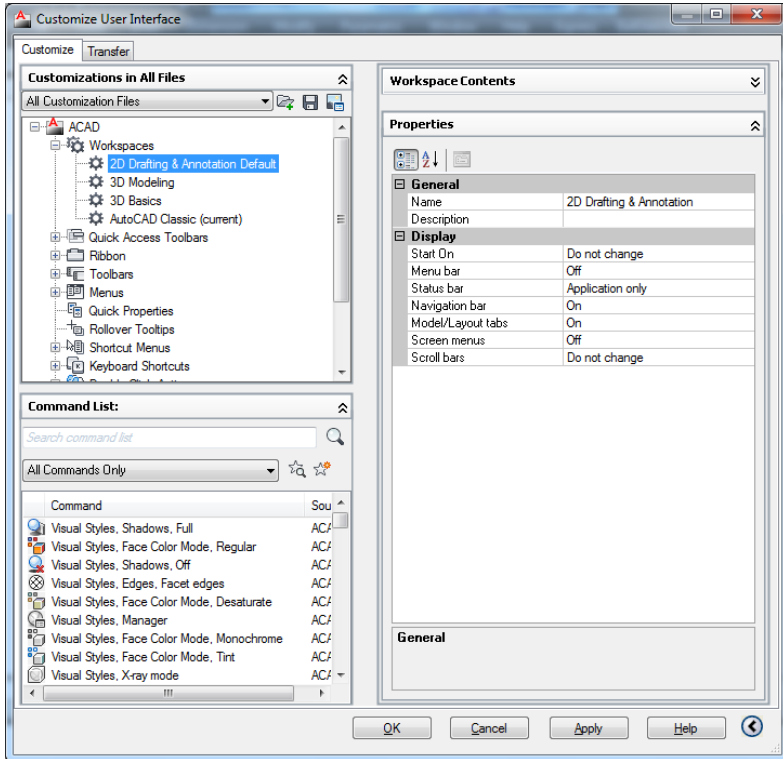


Рисунок 5.19 – Вікно Customize User Interface

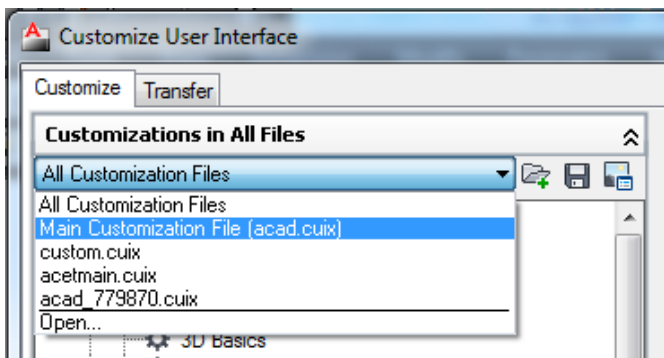


Рисунок 5.20 – Вибір файлу acad.cuix

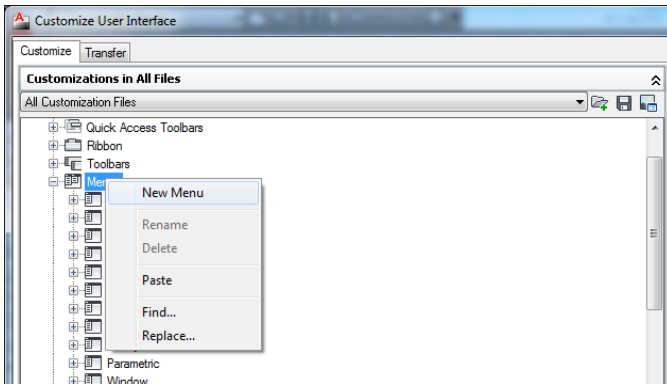


Рисунок 5.21 – Створення нового пункту меню

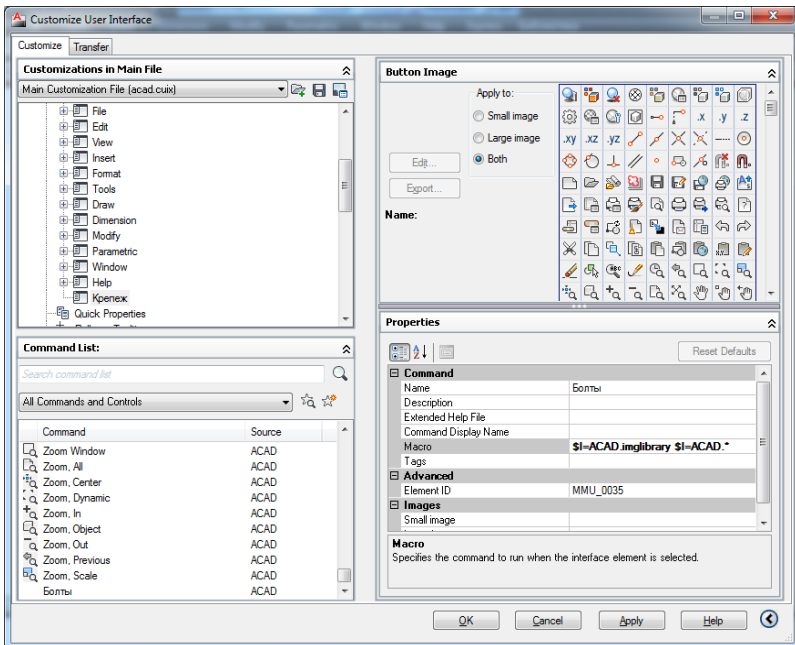


Рисунок 5.22 – Властивості нової команди Болты

Далі зі списку команд треба перетягнути нову команду **Болты** до нового меню **Крепезж** (див. рис. 5.23).

Натисніть кнопку *Save all current customizations files* (див. рис. 5.19). Закрийте гілку *Menus*.

Далі буде створене нове мозаїчне меню. Відкрийте гілку *Legacy* у структурі меню *ACAD*. У контекстному меню гілки *Image Tile Menus* оберіть пункт *New Image Tile Menu*.

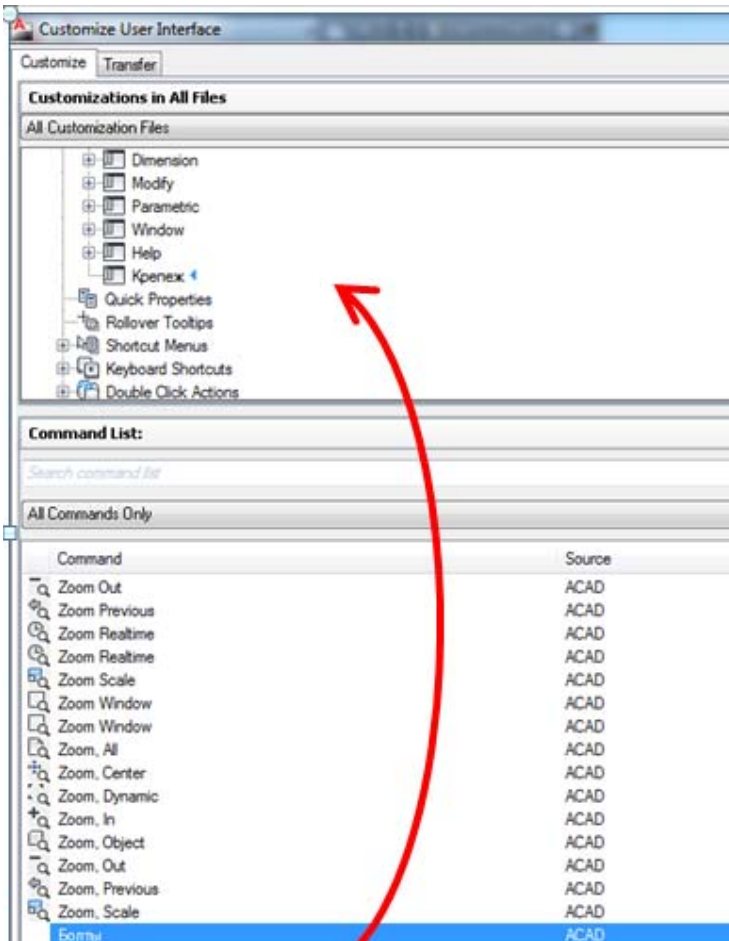


Рисунок 5.23 – Вставка нової команди Болты до меню Крепеж

Назвіть нове меню **Бібліотека**, та надайте йому псевдонім **IMGLIBRARY** (див. рис. 5.24).

Далі буде створена нова команда для мозаїчного меню. Перейдіть до розділу **Command List** та натисніть кнопку **Create a new command**. Назвіть команду **Болт ГОСТ 7798-70**, призначте макрос:

```
^C^C(load "test5.LSP") (GOST7798-70).
```

Далі зі списку команд треба перетягнути нову команду **Болт ГОСТ 7798-70** до нового мозаїчного меню **Бібліотека** (в випадку, якщо команда не буде перетягуватися в меню **Бібліотека**, перетягніть її спочатку в сусіднє меню, а потім безпосередньо в меню **Бібліотека**). Після цього серед властивостей команди з'являться дві нові – імена бібліотеки слайдів та слайду у бібліотеці (**Slide library**, **Slide label**). Назначте ім'я бібліотеки **lib7798**, ім'я слайду **779870m** (див. рис. 5.25).

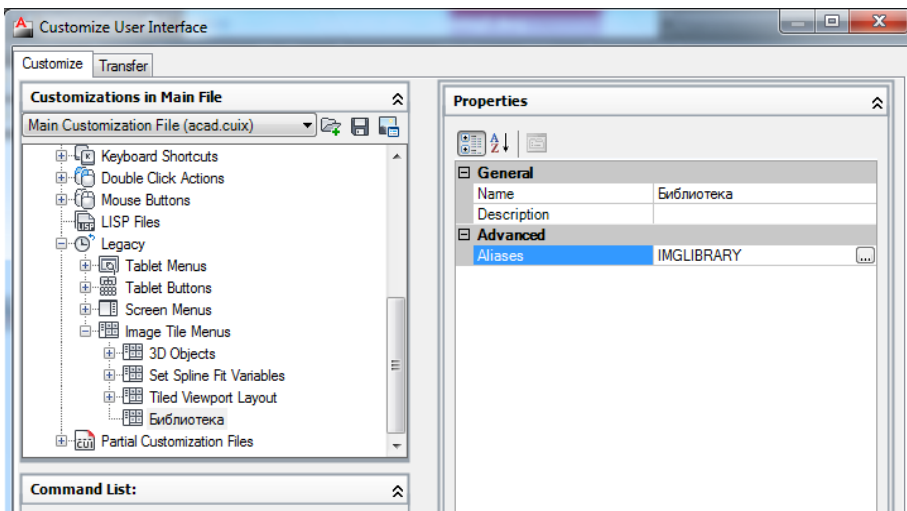


Рисунок 5.24 – Вставка нового мозаїчного меню та його властивості

Натисніть кнопку **Save all current customizations files**. Закрийте гілку **Legacy**.

Далі перейдіть на закладку **Transfer**. Необхідно перенести налаштування користувача до нового файлу. Знайдіть у дереві зліва щойно створені елементи та перетягніть їх направо до аналогічних розділів (див. рис. 5.26). У спадаючому списку замість **New File**

оберіть **Save As** та збережіть власні налаштування під будь-яким ім'ям (наприклад *NEW.CUIX*) до робочої папки.

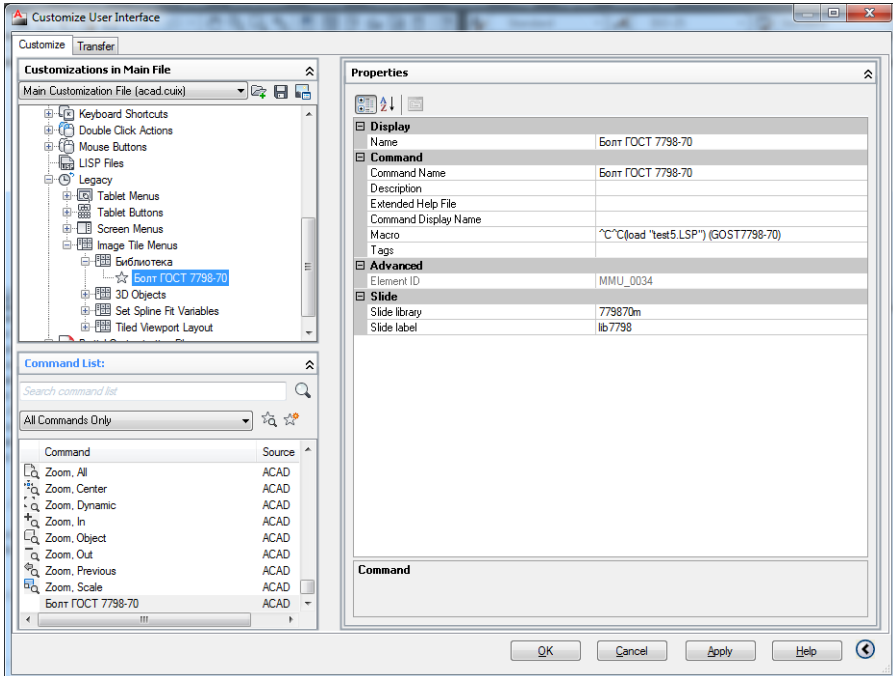


Рисунок 5.25 – Вставка нової команди до мозаїчного меню та її властивості

Перейдіть на закладку **Customize**. У спадаючому меню розділу **Customization in All Files** оберіть пункт **Open** та вкажіть на новий файл меню, який був щойно збережений. Натисніть **OK**.

Видаліть з файлу *ACAD.CUIX* усі зроблені налаштування.

У файлі *NEW.CUIX* виправте макрос, призначений на команду **Болты** (замініть назви груп меню з *ACAD* на *NEW*).

Знизу вікна **Customize User Interface** натисніть кнопки **Apply** та **OK**. Підключення завершено.

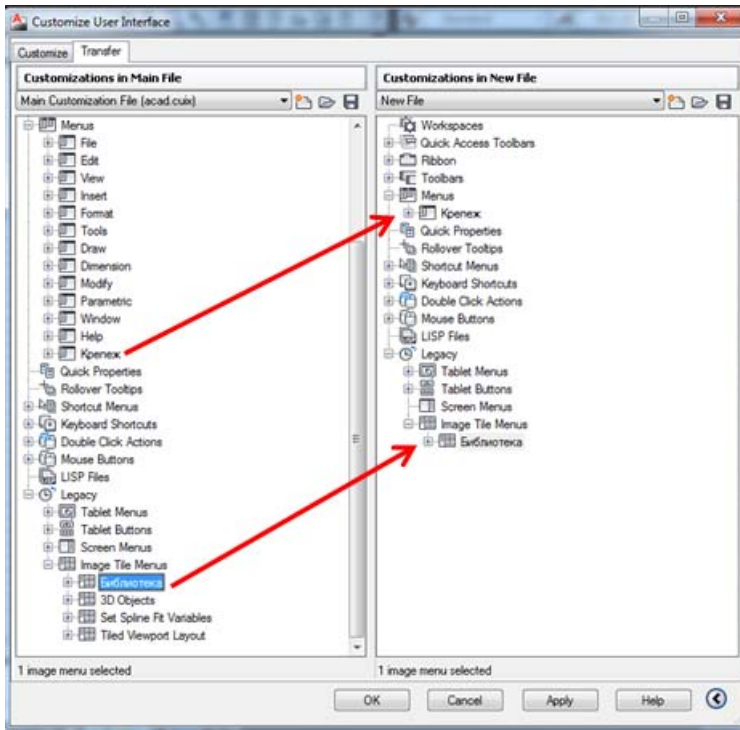


Рисунок 5.26 – Переміщення налаштувань до нового файлу

РЕКОМЕНДОВАНА ЛИТЕРАТУРА

1. Большаков, В.П. 3-D моделирование в AutoCAD, КОМПАС-3D, Solid Works, Inventor, T-Flex / В.П. Большаков, А.Л. Бочков, А.А. Сергеев. - СПб.: Питер, 2010.-336 с.
2. Autocad 2006: подробное иллюстрированное руководство. Под ред.. Жадаева А.Г..- М.: Лучшие книги, 2006.- 240 с.
3. Полешук Н.Н. Самоучитель AutoCAD 2000 и Visual LISP. – 2-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2001. – 672 с.: ил.
4. Юсупова М.Ф. Черчение в системе Autocad 2000. Учебное пособие.- К.: Алерта, 2003.-328с.
5. Свирневский Н.С. Моделирование и обработка изображений в среде AutoCAD 2000: Учебное пособие.- Хмельницкий: ТУП, 2000.- 360 с.
6. Автоматизация инженерно-графических работ. - Красильникова Г.А., Самсонов В.В., Тарелкин С.М.- СПб.: Питер, 2000. – 256с.: ил.
7. Романычева, Э.Т. Инженерная и компьютерная графика. Учебник для вузов.- 2е изд. перераб./ Э.Т.Романычева, Т.Ю.Соколова, Г.Ф. Шандурина М.:ДМК Пресс, 2001.-586 с.

Додаток А

Завдання на курсовий проект

Завдання – назви розділів бібліотеки КОМПАС, аналоги яких необхідно створити в середовищі AutoCAD.

Машиностроение

Конструкторская библиотека

ВИНТЫ

Винты нормальные

1. Винт ГОСТ 1491-80
2. Винт ГОСТ 11644-75
3. Винт ГОСТ 17473-80
4. Винт ГОСТ 17474-80
5. Винт ГОСТ 17475-80

Винты невыпадающие

6. Винт ГОСТ 10336-80
7. Винт ГОСТ 10337-80
8. Винт ГОСТ 10338-80
9. Винт ГОСТ 10339-80
10. Винт ГОСТ 10340-80
11. Винт ГОСТ 10341-80
12. Винт ГОСТ 10342-80
13. Винт ГОСТ 10343-80
14. Винт ГОСТ 10344-80

Винты установочные

15. Винт ГОСТ 1479-93
16. Винт ГОСТ 1481-84
17. Винт ГОСТ 1482-84
18. Винт ГОСТ 1483-84
19. Винт ГОСТ 1485-84
20. Винт ГОСТ 1486-84
21. Винт ГОСТ 1488-84

- 22. Винт ГОСТ 8878-84
- 23. Винт ГОСТ 11074-84
- 24. Винт ГОСТ 11075-93

Винты самонарезающиеся

- 25. Винт ГОСТ 10619-80
- 26. Винт ГОСТ 10620-80
- 27. Винт ГОСТ 10621-80
- 28. Винт ОСТ 37.001.181-81
- 29. Винт ОСТ 37.001.182-81
- 30. Винт ОСТ 37.001.184-81
- 31. Винт ОСТ 37.001.186-81
- 32. Винт ОСТ 37.001.188-81

БОЛТЫ

Болты нормальные

- 33. Болт ГОСТ 7798-70
- 34. Болт ГОСТ 7805-70
- 35. Болт ГОСТ 10602-94

Болты с полукруглой головкой

- 36 Болт ГОСТ 7802-81
- 37 Болт ГОСТ 7783-81
- 38 Болт ГОСТ 7801-81

Болты с потайной головкой

- 39 Болт ГОСТ 7785-81
- 40 Болт ГОСТ 7786-81
- 41 Болт ГОСТ 7787-81

Болты откидные

- 42. Рым-болт ГОСТ 4751-73

ГАЙКИ

Гайки шестигранные

Гайки нормальные

- 43. Гайка ГОСТ 5915-70,
- 44. Гайка ГОСТ 5927-70,
- 45. Гайка ГОСТ 15526-70

Гайки уменьшенные

- 46. Гайка ГОСТ 5916-70,
- 47. Гайка ГОСТ 5929-70,
- 48. Гайка ГОСТ 15521-70,
- 49. Гайка ГОСТ 15522-70,
- 50. Гайка ГОСТ 2524-70,
- 51. Гайка ГОСТ 2526-70

Гайки высокие

- 52. Гайка ГОСТ 5931-70,
- 53. Гайка ГОСТ 15523-70,
- 54. Гайка ГОСТ 15524-70,
- 55. Гайка ГОСТ 15525-70

Гайки прорезные и корончатые

- 56. Гайка ГОСТ 5918-73,
- 57. Гайка ГОСТ 5919-73,
- 58. Гайка ГОСТ 2528-73

Гайки шестигранные

Гайки круглые

- 59. Гайка ГОСТ 10657-80
- 60. Гайка ГОСТ 12460-67

ШАЙБЫ

61. Плоская шайба

- 62. Конич. и сферич. шайбы в сборе
- 63. Шайба ГОСТ 10906-78
- 64. Шайба ГОСТ 11648-75
- 65. Шайба ГОСТ 13438-68
- 66. Шайба ГОСТ 13439-68

67. Шайба ГОСТ 14734-69

Шайбы стопорные

68. Шайба ГОСТ 6402-70

69. Шайба ГОСТ 10462-81

70. Шайба ГОСТ 10463-81

71. Шайба ГОСТ 10464-81

72. Шайба ГОСТ 13463-77

73. Шайба ГОСТ 13464-77

74. Шайба ГОСТ 13465-77

ШУРУПЫ

75. Шурупы ГОСТ 1144-80

76. Шурупы ГОСТ 1145-80

77. Шурупы ГОСТ 1146-80

ШТИФТЫ

78. Штифты ГОСТ 3128-70

79. Штифты ГОСТ 3129-70

80. Штифты ГОСТ 9464-79

81. Штифты ГОСТ 9465-79

82. Штифты ГОСТ 10774-80

83. Штифты ГОСТ 12207-79

84. Штифты ГОСТ 19119-80

85. Штифты ГОСТ 12850.1-93

86. Штифты ГОСТ 12850.2-93

ШПОНКИ

87. Шпонки ГОСТ 23360-78

88. Шпонки ГОСТ 24071-97

89. Шпоночный паз ГОСТ 23360-78

90. Шпоночный паз ГОСТ 24071-80

Додаток Б

Функції LISP

SETQ – функції присвоювання.

Арифметичні функції:

+ – додавання будь-якої кількості чисел;

– – віднімання з першого числа суми будь-якої кількості чисел;

***** – перемножування будь-якої кількості чисел;

/ – ділення першого числа на добуток будь-якої кількості чисел;

1+ – збільшення числа на 1;

1– – зменшення числа на 1;

~ – доповнення числа до -1 ;

abs – абсолютна величина числа.

Операції порівняння:

= – перевірка на рівність;

/= – перевірка на нерівність;

< – перевірка на зростання зліва направо;

> – перевірка на спадання зліва направо;

<= – перевірка на неспадання зліва направо;

>= – перевірка на незростання зліва направо.

Функції, що виконують логічні операції:

and – операція логічного “І” для будь-якої кількості аргументів;

or – операція логічного “АБО” для будь-якої кількості аргументів;

not – операція логічного “Ні”;

null – операція перевірки на *nil*.

Контрольні конструкції:

If – умовна операція типу if-then-else (якщо-то-або);

prong – з’єднує вирази, якщо функція if використовується для виконання по умові не одного виразу, а декількох;

cond – умовна операція типу case з будь-якою кількістю умов;

while – дозволяє організовувати операції циклу по багатократній умові, що перевіряється;

Перевірочні функції:

minusp – перевірка чи є число від’ємним;

plusp – перевірка чи є число додатним;

evenp – перевірка чи є число парним;

oddr – перевірка чи є число непарним;
zerop – перевірка чи дорівнює число нулю;
numberp – перевірка чи є об'єкт числом;
listp – перевірка чи є вираз списком;
null – перевірка чи аргумент є пустим списком.

Обчислювані функції:

fix – скорочення числа до цілого;
rem – одержання залишку від ділення аргументів <число1> на <число2>, далі залишку від ділення результату на аргумент <число3> і т.п.;
max – визначення максимального з чисел;
min – визначення мінімального з чисел;
gcd – визначення найбільшого загального дільника для двох додатних цілих чисел;
angle – обчислення кута, в радіанах, між віссю X поточної системи координат та вектором, який іде з точки, заданої аргументом <точка1> у точку задану аргументом <точка2>;
distance – обчислення відстані між двома точками;
sqrt – обчислення квадратного кореня;
sin – обчислення синуса кута;
cos – обчислення косинуса кута;
atan – обчислення арктангенса;
exp – обчислення натурального антилогарифму – результату підведення у степінь постійної e (2.71828);
expt – підведення числа у степінь;
log – обчислення натурального логарифму.

Функції, що працюють зі списками та точковими парами:

list – створення списку;
append – злиття списків у один;
nth – витягання зі списку елемента по порядковому номеру (нумерація елементів списку виконується зліва направо та починається з нуля);
reverse – обернення списку;
car – витягання першого елемента списку або точкової пари;
cdr – витягання зі списку підсписку шляхом видалення першого елемента, якщо аргумент <список> є простим списком, або витягання другого елемента, якщо <список> є точковою парою;
last – виділяє останній елемент списку;
subst – замінює елемент у списку;

remove – видаляє елемент зі списку;

length – повертає у якості значення довжину списку.

Cons – додає до списку перший елемент або створює точкову пару;

member – перевіряє належність елемента списку;

assoc – здійснює пошук пари по відповідному ключу у асоціативному списку;

apply – застосування функції, заданою аргументом <функція> по черзі до всіх елементів списку, заданого аргументом <список>.

Mapcar – застосування функції, заданою аргументом <функція> спочатку до перших елементів кожного зі списків (<список1>, ..., <списокN>), далі – до других елементів і т.п.; результати об'єднуються у новий список, який є значенням, яке повертається.

Foreach – дозволяє організувати циклічне виконання виразів, по черзі підставляючи замість змінної циклу елементи списку.

Додаток В

Функції AutoLISP

LOAD – функція завантаження файлу.

COMMAND – забезпечує доступ до команд AutoCAD.

Читання та задання системних змінних AutoCAD у AutoLISP:

getvar – визначає значення системної змінної AutoCAD у дану мить;

setvar – присвоює системній змінній AutoCAD певне значення;

getenv – здійснює читання поточних значень змінних, заданих в середовищі операційної системи.

Функції геометричних обчислень:

angle – обчислює кут (у радіанах) між двома променями, які виходять з точки <точ.1> (перший промінь – вздовж позитивного напрямку вісі X, інший – у бік точки <точ.2>);

distance – обчислює відстань між двома двовимірними або тривимірними точками;

inters – обчислює точку перетину двох відрізків (якщо нема пересічення - nil);

polar – визначає точку, яка знаходиться під кутом (у радіанах) та на відстані від заданої точки <точ.1>;

osnap – визначає точку, яка є результатом застосування об'єктної прив'язки.

Функції організації введення даних різного типу:

getangle, getorient – здійснюють введення кута у радіанах;

getpoint, getcorner – здійснюють введення точки;

getdist – здійснює введення відстані;

getint – здійснює введення цілого числа;

getkword – здійснює введення ключового слова;

getreal – здійснює введення дійсного числа;

getstring – здійснює введення рядка тексту.

Функції імені примітива:

entnext – одержує ім'я наступного примітива;

entlast – одержує ім'я останнього невидаленого примітива;
entsel – вибирає примітив.

Функції маніпулювання набором примітивів:

ssget – отримує набір примітивів;
sslength – обчислює кількість примітивів в наборі;
ssname – отримує ім'я примітиву з набору;
ssadd – додає примітив в набір;
ssdel – видаляє примітив з набору;
ssmemb – перевіряє наявність примітива в наборі.

Функції оперування даними примітивів креслення:

entdel – видаляє примітив;
enttget – отримує інформацію про примітив;
entmod – модифікує інформацію про примітив;
entupd – оновлює зображення на екрані;
entmake – створює новий примітив в базі даних рисунка.

Функції перетворення тупів:

rtos – перетворює дійсне число у рядок;
distof – виконує дію, протилежну дії *rtos*.
angtos – перетворює кут у рядок;
angtof – виконує дію, протилежну дії функції *angtos*;
cvunit – переводить величину або координати точки з одних одиниць виміру у інші.

Функція перетворення систем координат:

trans – перетворює точки та відстані з одної системи координат у іншу.

Функції керування екраном:

prompt – видає повідомлення у зоні підказок екрану та повертає nil;
print – призначена для виводу S-виразу на екран або у файл та повертає S-вираз з нового рядка наступним пропуском;
grtext – дозволяє писати у певних частинах графічного екрану AutoCAD.